Deutsches Zentrum
für Luft- und Raumfahrt e.V.

3D-Bauteilmanipulation zur Satellitenkonfiguration unter Verwendung von Mixed-Reality-Technik

Artur Baranowski

B. Eng. Medientechnik
Hochschule Düsseldorf
Fachbereich Medien

27. März 2018

Betreuung
Prof. Dr. Eng. / Univ. of Tsukuba Jens Herder

Externe Betreuung
M. Sc. Sebastian Utzig

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Sämtliche Stellen der Arbeit, die im Wortlaut oder dem Sinn nach fremden Quellen entnommen sind, habe ich als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und noch nicht veröffentlicht.

Ort, Datum	Unterschrift

Abstract

The development of a spacecraft is a complex endeavour, requiring expertise in numerous different domains, such as mathematics, physics and engineering. Scientists and engineers are facing the challenge of explaining their specialized knowledge to experts of other subject areas. To communicate the specific constraints of their respective subsystem appropriately, creative communication methods, such as 3D visualizations are necessary. Recent display technologies developed for Augmented and Virtual Reality offer new ways of presenting computer-generated content. Additionally, it enables the direct interaction with the virtual content. By means of a practical implementation the possibilities and challenges for spacecraft configuration employing this technology are explored and discussed. Using an optical see-through Head-Mounted Display an entire interaction concept for a 3D user interface is developed. Finally, a user study is conducted to evaluate the usability of the outlined user interface. Advantages and disadvantages of AR technology in combination with spacecraft configuration as well as potentials for enhancement of the proposed user interface are revealed.

Keywords: spacecraft configuration, human computer interaction, augmented reality, 3D user interfaces

Zusammenfassung

Der Bau eines Raumfahrzeugs ist ein komplexer Prozess, der Expertenwissen aus zahlreichen, unterschiedlichen Domänen der Physik, Mathematik und Ingenieurswissenschaften erfordert. Dabei stehen die Akteure vor der Herausforderung, Experten fachfremder Bereiche ihr spezialisiertes Wissen zu vermitteln. Um die spezifischen Rahmenbedingungen ihrer Subsysteme sachgerecht kommunizieren zu können, greifen Sie auf kreative Methoden zurück und stellen das Raumfahrzeug mit Hilfe von 3D-Visualisierungen dar. Moderne Anzeigetechnologie aus den Bereichen Augmented und Virtual Reality ermöglichen eine neuartige Darstellung computergenerierter Inhalte. Sie ermöglichen dabei eine natürliche Interaktion mit den Inhalten der virtuellen Umgebungen. Die Anforderungen der Raumfahrzeugmontage unter Verwendung von AR-Hardware werden in dieser Arbeit anhand einer praktischen Implementierung untersucht. Mit Hilfe eines optischen Head-Mounted Displays wird dazu ein vollständiges Interaktionskonzept für eine 3D-Benutzerschnittstelle entwickelt. Um die Gebrauchstauglichkeit der entwickelten Benutzerschnittstelle zu untersuchen wird sie schließlich in einer Nutzerstudie evaluiert. Vor- und Nachteile bei Verwendung von AR-Technologie während der Raumfahrzeugmontage werden aufgedeckt und Verbesserungspotentiale offengelegt.

Keywords: Raumfahrzeugbau, Mensch Computer Interaktion, Augmented Reality, 3D Benutzerschnittstellen

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Das folgende Kapitel führt in die Thematik dieser Arbeit ein. Abschnitt 1.1 vermittelt die Beweggründe für die Anfertigung der Arbeit. Anschließend wird in Abschnitt die Zielsetzung der Arbeit formuliert und ein inhaltlicher Überblick gegeben.

1.1 Motivation

Der Bau eines Raumfahrzeugs ist ein komplexes Unterfangen, welches die Zusammenarbeit eines interdisziplinären Teams erfordert. Die einzelnen Experten bringen hierbei domänenspezifisches Wissen innerhalb eines Projekts ein und arbeiten an einzelnen Subsystemen. Zwischen den Subsystemen bestehen gleichzeitig viele Abhängigkeiten. Daher ist zuverlässige und effiziente Kommunikation ein integraler Bestandteil beim Raumfahrzeugentwurf. Mit dem Ziel, die Kommunikation zwischen Experten eines Teams zu unterstützen, richtete das Deutsche Zentrum für Luft und Raumfahrt (DLR) 2008 in Bremen die Concurrent Engineering Facility (CEF) ein [1]. Sie ermöglicht Ingenieuren unterschiedlicher Disziplinen, kollaborativ eine Raumfahrtmission zu konzeptualisieren, dabei Abhängigkeiten zu erkennen und auftretende Konflikte effizient zu lösen (siehe Abbildung 1.1).

Obwohl die CEF Austausch in wesentlichen Aspekten unterstützt, behindern domänenspezifische Arbeitsmittel weiterhin den Datentransfer zwischen den Experten [2]. Jede Disziplin erstellt ein eigenes, spezifisches Modell für den Systementwurf. Hierbei nutzen sie gegebenfalls auch domänenspezifische Werkzeuge für die Entwicklung ihrer Subsysteme. Ein Strukturingenieur nutzt beispielsweise spezielle CAD[1]-Software für den Entwurf des Tragwerks eines Raumfahrzeugs, während ein Ingenieur, der Attitude and Orbital Control Systems (AOCS) entwickelt, Software-Werkzeuge für Orbit Kalkulationen und Analysen verwendet [2]. Um die dieses Problem zu überwinden, entwickelte die Einrichtung für Simulations- und Softwaretechnik am DLR ein Software-Tool namens *Virtual Satellite* (siehe Abschnitt 2.3), das nach der Methode des Model Based Systems Engineerings (MBSE) operiert. MBSE ist eine Weiterentwicklung dokumentenbasierter Systems-Engineering-Ansätze mit Datenmodellen, wie sie beispielsweise von der European Space Agency (ESA) in Form des Integrated Design Models (IDM) genutzt wurden [3]. Virtual Satellite verfügt über ein für alle relevanten Domänen des Raumfahrzeugbaus konsistentes

[1] computer aided design

Datenmodell, das von den spezifischen Werkzeugen und Methoden einzelner Disziplinen abstrahiert. So kann das Systems Engineering domänenübergreifend in computergerechte Arbeitsprozesse umgewandelt werden, die sich automatisieren lassen.

Wissenschaftler und Ingenieure stehen jedoch weiterhin vor der Herausforderung, Experten anderer Disziplinen ihr spezialisiertes Wissen zu kommunizieren. Um Verständnisprobleme zu überwinden und ihr fortgeschrittenes Wissen erläutern zu können, greifen Ingenieure auf kreative Kommunikationsmethoden zurück. Erfahrungen aus der Arbeit im CEF zeigen, dass Ingenieure spezielle, domänenspezifische Sachverhalte mithilfe von Gesten und Skizzen erklären [2]. Um diese Beobachtungen aufzugreifen, erweiterte die Einrichtung für Simulations- und Softwaretechnik am DLR das MBSE-Tool Virtual Satellite um eine auf dem Vizualization Toolkit (VTK) basierende Visualisierungskomponente [4]. Diese ermöglicht die Darstellung des Raumfahrzeugs und seiner Subsysteme in einem virtuellen 3D-Modell (siehe Abbildung 1.2).

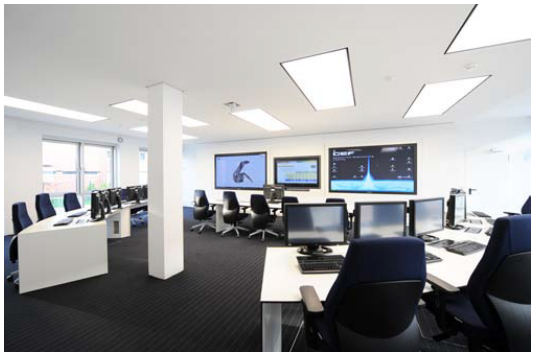


Abbildung 1.1: Concurrent Engineering Facility des DLR in Bremen [2].

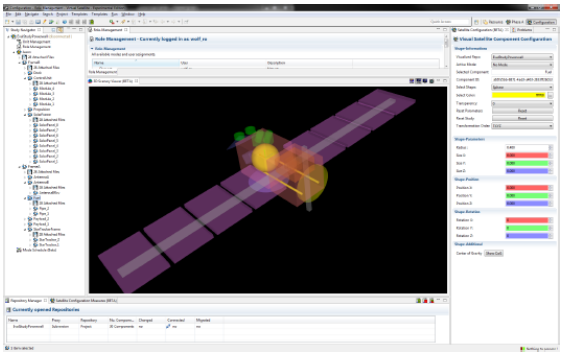


Abbildung 1.2: Virtual Satellite GUI mit VTK-Visualisierungskomponente [2].

Die voranschreitende Entwicklung stereoskopischer Anzeigegeräte in den Bereichen Virtual Reality (VR) und Augmented Reality (AR) ermöglicht einen neuartigen Zugang zu virtuellen Umgebungen. Eine besondere Klasse von Anzeigegeräten sind hierbei Head-Mounted-Displays (HMDs). Sie geben virtuelle Inhalte über zwei augennahe Displays bzw. Projektionsflächen wieder. Dadurch nimmt der Betrachter virtuelle Objekte dreidimensional im Raum wahr und erhält den Eindruck, sich in der computergenerierten Bildlandschaft zu befinden (*Immersion*, vgl. Abschnitt 2.1). Im Gegensatz dazu birgt die desktopbasierte Visualisierungskomponente von Virtual Satellite Gefahren der Fehlinterpretation seitens des Betrachters, da dreidimensionale Inhalte auf eine zweidimensionale Oberfläche projiziert werden müssen. Ware et al. konnten zeigen, dass eine stereoskopische Darstellung dreidimensionaler, virtueller Inhalte die räumliche Wahrnehmung im Vergleich zu zweidimensionalen Bildschirmoberflächen verbessert [5]. Unter Berücksichtigung des Blickwinkels eines Nutzers können virtuelle Objekte mit Hilfe von HMDs auf natürliche Weise aus unterschiedlichen Blickrichtungen betrachtet werden. Gleichzeitig ermöglichen neuartige Benutzerschnittstellen für immersive Anwendungen eine unmittelbare Interaktion mit den virtuellen Komponenten. Durch geeignete Eingabemechanismen, wie der Sprach- oder Gestensteuerung, können virtuelle 3D-Modelle im Raum positioniert und gedreht werden.

Im Zusammenhang mit der Raumfahrzeugentwicklung und insbesondere den Arbeitsprozessen in der CEF hat immersive AR-Hardware gegenüber vergleichbarer VR-Hardware folgende Vorteile: Anwender nehmen weiterhin die physisch-reale Umwelt wahr. Dadurch bleibt die natürliche Kommunikation zwischen Experten mittels Körpersprache, Gestik und Mimik erhalten. Dies ist entscheidend, denn die Verbesserung der Kommunikation zwischen Ingenieuren unterschiedlicher Domänen ist eine zentrale Motivation für die CEF. Zusätzlich kann weiterhin die klassische Virtual Satellite Desktop-Schnittstelle in Kombination mit der Anzeigeschnittstelle des HMD genutzt werden.

Beispiele für immersive AR-HMDs sind die *HoloLens* von Microsoft oder *Google Glass* [6, 7]. Auf dem Gebiet der AR-Technik findet rege und aktive Entwicklungsarbeit statt, wie die sich in der Entwicklung befindlichen Geräte *Magic Leap One* oder *Vaunt* von Intel zeigen [8, 9]. Unter den verfügbaren AR-HMDs stellt die HoloLens vergleichsweise weit fortgeschrittene und anspruchsvolle Hardware dar. Sie verfügt über leistungsfähige Prozessortechnik, ist dabei nicht kabelgebunden und vollständig mobil (vgl. Abschnitt 2.2). Innerhalb der integrierten Entwicklungsumgebung Visual Studio und der Spiele-Engine Unity werden zudem spezielle Software-Werkzeuge zur Verfügung gestellt, welche die Entwicklung und Bereitstellung von HoloLens-Applikationen wesentlich erleichtern.

1.2 Ziel und Struktur

Um die Vor- und Nachteile immersiver AR-Applikationen bei der Konzeption von Raumfahrzeugen in der CEF untersuchen zu können, wurde im Rahmen dieser Arbeit eine Anwendung für die Microsoft HoloLens (vgl. Abschnitt 2.3) entwickelt. Die Anwendung ist eingebettet in eine Client-Server-Architektur und ist in der Lage, Visualisierungsdaten mit Serverinstanzen von Virtual Satellite auszutauschen. Aufgaben der Clientanwendung sind der Empfang, die Darstellung und die Manipulation der Visualisierung eines von Virtual Satellite erzeugten Systemmodells. Der Schwerpunkt liegt dabei auf der Entwicklung eines ganzheitlichen Interaktionskonzepts für die Interaktionsmetapher, wie sie durch die Eingabeschnittstelle der Microsoft HoloLens ermöglicht wird (*Natural User Interfaces*, vgl. Abschnitt 2.1). Insbesondere die effektive und intuitive Manipulation der Position und Orientierung von virtuellen Objekten steht hierbei im Fokus.

Kapitel 2 dieser Arbeit befasst sich mit den Grundlagen und vermittelt notwendige Voraussetzungen für die Entwicklung eines Interaktionskonzepts für die 3D-Objektmanipulation. Zunächst werden die Begrifflichkeiten Mixed Reality und Augmented Reality definiert, um die Microsoft HoloLens von anderen AR-Techniken abzugrenzen. Anschließend werden technische Besonderheiten der Microsoft HoloLens erörtert. Da die Eignung einer Eingabemetapher in wesentlichen Teilen von den technischen Rahmenbedingungen des Eingabesystems abhängt, wird insbesondere auf die verfügbaren Eingabemechanismen und -schnittstellen eingegangen. Schließlich wird das MBSE-Tool Virtual Satellite und die dafür entwickelte Clientanwendung vorgestellt. Im Besonderen wird auf die Client-Server-Architektur und die Kommunikationsschnittstelle zwischen den Anwendungen eingegangen.

Kapitel 3 befasst sich anschließend mit der Entwicklung des Interaktionskonzepts für die HoloLens-Anwendung. Zunächst wird hierzu der Nutzungskontext detailliert aufgeschlüsselt. Auf dieser Grundlage werden Anforderungen formuliert, welche das Konzept bedienen muss. Dann wird die Interaktion in virtuellen Umgebungen, gemäß einer Taxonomie von Bowman et al. [10], in vier Teilaspekte zerlegt und implementiert. Der erste Abschnitt befasst sich mit der Systemsteuerung. Die folgenden beiden Abschnitte behandeln anschließend die eigentlichen Selektions- und Manipulationstechniken, welche eine Interaktion mit den virtuellen Systemkomponenten möglich machen.

Das implementierte Interaktionskonzept wird im Anschluss in Kapitel 4 durch eine Nutzerstudie evaluiert. Dabei wird die Benutzerschnittstelle der HoloLens-Anwendung hinsichtlich diverser quantitativer und qualitativer Messgrößen mit Virtual Satellite als Referenz-Schnittstelle verglichen. Die Ergebnisse der Arbeit werden schließlich ausgewertet und in einem Fazit zusammengefasst.

2 Grundlagen

Bevor mit der Entwicklung eines Interaktionskonzepts begonnen werden kann, muss eine Clientapplikation für die HoloLens entwickelt werden. Erst wenn Visualisierungsmodelle auf der HoloLens empfangen und dargestellt werden können, ist eine Interaktion mit diesen möglich. Hierzu wird die Microsoft HoloLens in den Abschnitten 2.1 und 2.2 zunächst in den breiteren Kontext von Mixed Reality eingeordnet und hinsichtlich ihrer technischen Rahmenbedingungen untersucht. In den Abschnitten 2.3 und 2.4 wird auf Virtual Satellite eingegangen und die in dessen Client-Server-Architektur eingebettete HoloLens-Clientanwendung erörtert.

2.1 Mixed Reality

Sowohl in der Wirtschaft als auch in der Wissenschaft gewinnen *Virtual Reality* (VR) und *Augmented Reality* (AR) zunehmend an Aufmerksamkeit. Die Anwendungsfälle reichen von der Medizintechnik bis zum Marketing [11, 12]. Trotz etablierter Definitionen werden diese, sowie verwandte Begrifflichkeiten wie *Mixed Reality* oder *Augmented Virtuality*, nicht selten missverstanden. So vermarktet beispielsweise Microsoft die HoloLens als Mixed-Reality-Brille [6].

Milgram et al. [13] führten das Reality-Virtuality Kontinuum ein und ermöglichen eine differenzierte Betrachtung der Begrifflichkeiten. Hierbei werden Anwendungen hinsichtlich der präsentierten Umwelt entlang eines eindimensionalen Kontinuums eingeordnet (siehe Abbildung 2.1). Die rein physisch-reale Welt ist auf dem Kontinuum links zu verorten, wohingegen eine Anwendung mit rein virtueller Umwelt rechts zu verorten ist. Dazwischen befinden sich Anwendungen, in denen die physisch-reale Umwelt mit einer virtuellen Umwelt vermengt wird. In diesem Zusammenhang sprechen Milgram et al. von Mixed Reality. Abhängig vom Verhältnis zwischen physisch-realen und virtuellen Bestandteilen in MR-Anwendungen wird weiter zwischen Augmented Reality und Augmented Virtuality differenziert. Bei AR-Anwendungen besteht der überwiegende Teil der wahrgenommenen Umwelt aus physisch-realen Elementen, wohingegen bei AV-Anwendungen die virtuelle Umwelt dominanter ist.

Das RV-Kontinuum ermöglicht eine erste, grobe Differenzierung. Insbesondere jedoch für AR- bzw. MR-Applikationen ist des Weiteren eine Unterscheidung zwischen immersiven und nicht-immersiven Anwendungen zu treffen. Von Immersion spricht

man, wenn die simulierten Stimuli vom Nutzer als Bestandteil der realen Umwelt wahrgenommen werden. Der Nutzer nimmt sich dabei ebenfalls als Bestandteil der virtuellen Umwelt wahr und interagiert mit ihr [13]. Im weitesten Sinne ist bereits eine Fernsehübertragung mit CG-Overlays als ein Fall von AR aufzufassen [13]. Diese und andere Techniken, mit denen virtuelle Inhalte mit Echtwelt-Bildern auf monoskopischen Anzeigegeräten vermengt werden sind deutlich von immersiven Anzeigetechniken abzugrenzen, die eine stereoskopische Wahrnehmung virtueller Inhalte im Raum ermöglichen. Zudem ist eine Differenzierung zwischen AR-Anwendungen, die virtuelle Objekte in Beziehung zum Raum setzen können (engl. *Spatial Awareness* bzw. *Spatial Mapping*, vgl. Abschnitt 2.2.1), und Anwendungen, die virtuelle Inhalte lediglich auf die physisch-reale Welt projizieren, notwendig. Die Microsoft HoloLens ist in diesem Zusammenhang als immersive AR-Hardware zu klassifizieren. Die physisch-reale Umwelt ist durch einen transparenten Sichtschirm vollumfänglich wahrnehmbar und wird durch augennahe Projektionsflächen mit virtuellen Objekten angereichert (vgl. Abschnitt 2.2).

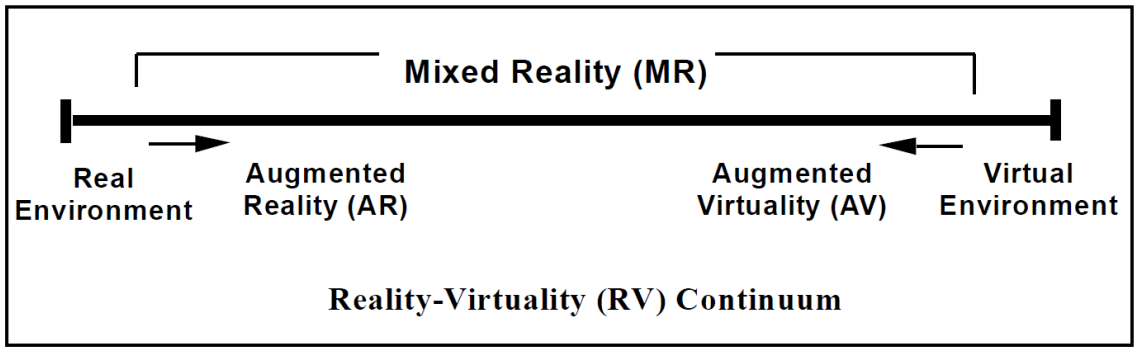


Abbildung 2.1: Vereinfachte Darstellung des RV-Kontinuums [13].

Ein weiterer, wesentlicher Bestandteil immersiver AR- und VR-Hardware ist die Anwendung sogenannter *Natural User Interfaces* (NUI) [14]. Anders als graphische Benutzerschnittstellen, die häufig auf dem WIMP (kurz für „window, icon, menu, pointer“) Grundkonzept basieren [15], bauen NUIs stark auf das implizite Wissen des Anwenders über die alltägliche, reale Welt auf [14]. Bei der Interaktion wird der eigene Körper oder das Verständnis des Anwenders über grundlegende physikalische Zusammenhänge genutzt, um die Benutzerschnittstelle leicht zugänglich und erlernbar zu machen. So imitieren NUIs die Interaktion in der realen, nicht-digitalen Welt. Auch das Eingabesystem der HoloLens basiert mit Techniken wie der Gesten- oder Sprachsteuerung auf NUIs (vgl. Abschnitt 2.2.1).

2.2 Microsoft HoloLens

Für die praktische Umsetzung wird die Microsoft HoloLens als Darstellungs- und Interaktionsmedium verwendet. Hierbei handelt es sich um ein optisches Head-Mounted-Display mit transparentem Sichtschirm[1]. Die Eignung bestimmter Interaktionsmetaphern und -techniken hängt in wesentlichen Teilen von den technischen

[1] engl. *optical see-through Head-Mounted Display*

Rahmenbedingungen der HoloLens ab. Daher lohnt sich eine genauere Betrachtung der Anzeige- und Eingabemechanismen der HoloLens.

Das Kernelement der HoloLens ist ein Cherry Trail SoC[2] von Intel. Zusammen mit der sogenannten *Holographic Processing Unit* (HPU) stellt es die zentrale Stelle für die Verarbeitung von Daten dar. Die HPU ist ein eigens für die HoloLens entwickelter Prozessor zur effizienten Entgegennahme und Aufbereitung sämtlicher durch die Sensoren der HoloLens erfassten Daten. Die von der HPU produzierten Datenpakete werden anschließend zur Weiterverarbeitung an das SoC weitergeleitet [16]. Das SoC basiert auf einer x86-Prozessorarchitektur, welches durch ein mobiles Windows 10 Betriebssystem verwaltet wird. Es übernimmt die Steuerung des Betriebssystems und der auf diesem laufenden Applikationen. Eine Zusammenfassung der erwähnten und weiterer Spezifikationen können Tabelle 2.1 entnommen werden.

Prozessor	Intel x86-Prozessorarchitektur, HPU 1.0
Konnektivität	Wi-Fi 802.11ac, Micro-USB 2.0, Bluetooth 4.1 LE
Speicher	64 Gigabyte
RAM	2 Gigabyte
Video	2 Megapixel Foto-, HD-Videokamera
Audio	3,5mm Klinkenstecker, externe Lautsprecher
Gewicht	579 Gramm

Tabelle 2.1: Hardware Spezifikationen der Microsoft HoloLens [17].

Die Standard-Laufzeitumgebung für alle HoloLens Applikationen ist die Universal Windows Platform (UWP). Auf Basis von UWP entwickelte Applikationen sind unter einer Vielzahl von Microsoft-Zielgeräten mit wenig bzw. keinem zusätzlichen Portierungsaufwand lauffähig.

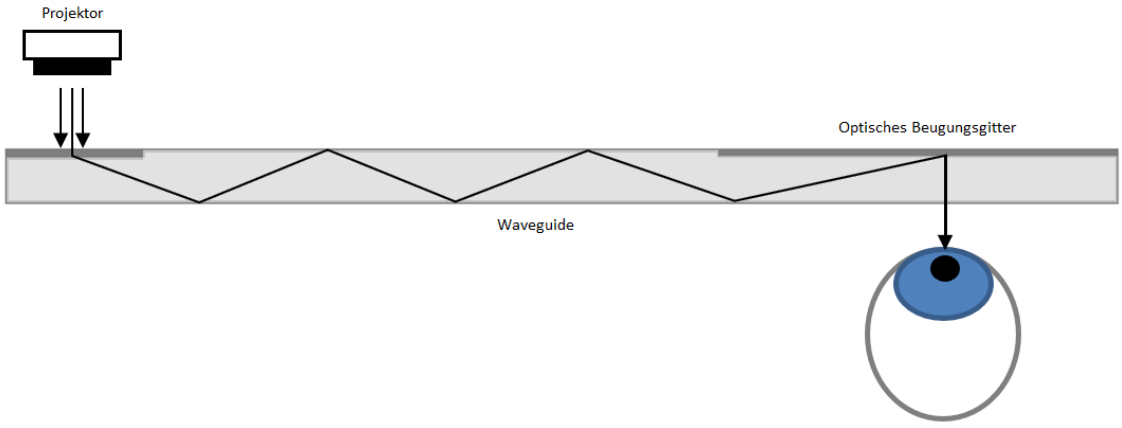


Abbildung 2.2: Prinzipdarstellung des optischen Systems der HoloLens [18].

Zur Projektion der virtuellen Inhalte nutzt die HoloLens zwei LCoS[3]-Displays die am Linsenbügel angebracht sind und Licht entlang planarer Lichtwellenleiter[4]

[2] System-on-a-Chip, dt. Ein-Chip-System
[3] Liquid Crystal on Silicon, dt. Flüssigkristalle auf Siliziumsubstrat
[4] engl. *waveguide*

emittieren (vgl. Abbildung 2.2). Um das Bild aus den Lichtwellenleitern extrahieren zu können, werden optische Beugungsgitter an den entsprechenden Stellen der Oberfläche des Lichtwellenleiters angebracht. Das Beugungsgitter erzeugt ein Interferenzmuster, welches das Licht aus dem Lichtwellenleiter herausbeugt und es dabei vergrößert. Dieses Verfahren bezeichnet man als Exit Pupil Expansion[5]. Der Bereich, der den Lichtwellenleiter mit der Linse und dem Beugungsgitter kombiniert, bezeichnet man als Combiner. Der Sichtbereich der HoloLens hat eine Größe von $30 \times 17,5$ Grad . Dort wird das auf dem Auge auftreffende Licht aus der physischen Umwelt mit der Projektion der LCoS-Displays überlagert. Weil durch die beiden Displays separate Bilder für jedes Auge projiziert werden, kann das aus dem Lichtwellenleiter extrahierte Bild dreidimensional im Raum wahrgenommen werden.

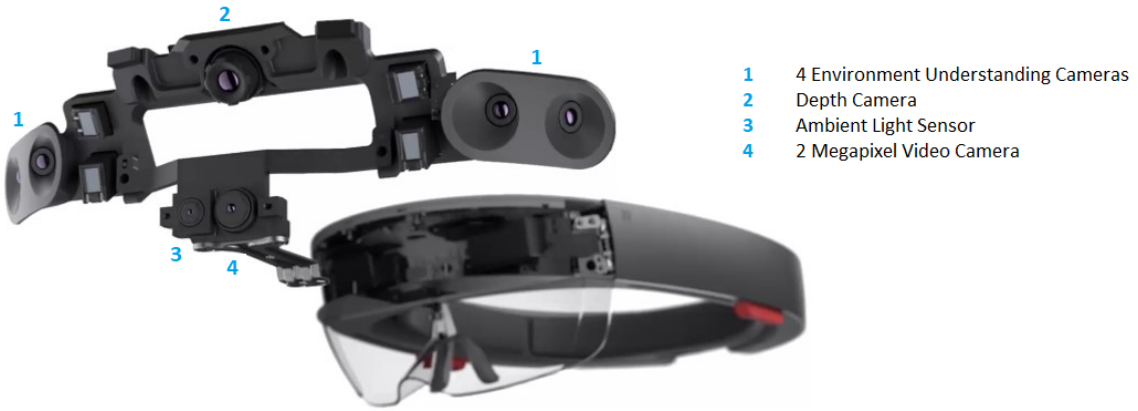


Abbildung 2.3: Sensoren am vorderen Bügel der HoloLens [17].

Grundlegend erfassen an der HoloLens angebrachte Sensoren jedgliche Benutzereingaben. Der Großteil der Sensoren ist an einer Leiste angebracht, die in den Bügel an der Vorderseite der HoloLens eingelassen ist (siehe Abbildung 2.3). Darunter befinden sich vier sogennante „Umwelt erfassende“ Kameras[6]. Sie dienen der Lokalisierung der Kopfposition im Raum [19]. Eine Time-of-Flight Kamera ermöglicht das Tracking der Hände des Anwenders und die virtuelle 3D-Rekonstruktion des Raumes in Echtzeit (*Spatial Mapping*, vgl. Abschnitt 2.2.1). Sie verfügt über ein 120×120 Grad breites Sichtfeld. Mit Hilfe einer inertialen Messeinheit[7] (IMU) kann unter anderem die Orientierung der HoloLens im Raum bestimmt werden. Die HoloLens verfügt weiterhin über einen Umgebungslichtsensor, vier Mikrofone für die Sprachsteuerung sowie eine zwei Megapixel Videokamera.

2.2.1 Eingabeschnittstelle

Mithilfe der Sensoren sind drei grundlegende Eingabeformen gebrauchsfertig nutzbar. Die Eingabeformen sind die Blickrichtung („Gaze“), sowie die Gesten- und Sprachsteuerung. Durch die Gaze werden virtuelle Objekte, Icons oder Widgets fokussiert. So wird ein Bezug zwischen Aktionen, ausgelöst durch Gesten- oder Sprachbefehle, und virtuellen Objekten hergestellt. Um die Gestensteuerung zu aktivieren, muss sich die Hand des Anwenders innerhalb des Sichtfelds der ToF-Kamera und in

[5] dt. Expansion der Austrittspupille
[6] engl. *environment understanding cameras*
[7] engl. *inertial measurement unit*

der sogenannten „Ready“-Pose befinden (vgl. Abbildung 2.4). Durch das Beugen des Zeigefingers zum Daumen und Zurückkehren in die Ready-Pose kann ein sogenannter „Air-Tap“ durchgeführt werden. Dieser Bewegungsablauf ist einem Mausklick nachempfunden und entspricht einem einzelnen, diskreten Befehl zur Validierung von Aktionen bzw. Selektionen [20]. Zwei unmittelbar nacheinander durchgeführte Air-Taps entsprechen einer „Double-Tap“-Geste. Durch Neigen des Zeigefingers zum Daumen und Verharren in dieser Position, kann eine „Tap-and-Hold“-Geste durchgeführt werden. Hierdurch werden kontinuierliche Manipulationen ermöglicht, wie etwa das Verschieben eines virtuellen Objekts.

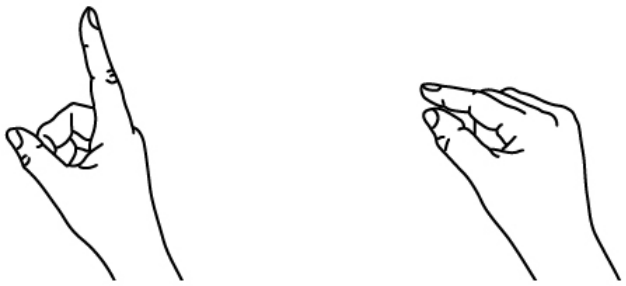


Abbildung 2.4: Ready-Pose (links) und Tap-Geste (rechts) [20].

Gaze und Gestensteuerung, sowie die die von der ToF-Kamera erfassten Parameter der Handposition können für die Entwicklung von HoloLens-Applikationen praktisch genutzt werden. Hierfür wird innerhalb der UWP Laufzeitumgebung eine Programmierschnittstelle[8] (API) für die Entwicklung von MR-Applikationen bereitgestellt. Aufbauend auf der API entwickelte Microsoft das *Mixed Reality Toolkit* (MRTK) und eine Portierung dessen für die Entwicklung von Anwendungen mit Unity3D. Bei Unity3D handelt es sich um eine plattformunabhängige Entwicklungsumgebung für Spiele (Game-Engine), die von Unity Technologies entwickelt wird. Sie ermöglicht die Entwicklung von 3D-Grafik-Anwendungen mit Hilfe der Skriptingsprache C-Sharp. Das MRTK erleichtert die Entwicklung von UWP Applikationen für die HoloLens und anderer immersiver MR-Headsets. Insbesondere gewährt es Zugriff auf die Eingabeparameter durchgeführter Gesten, der Handposition und der Blickrichtung. Eine deutliche Einschränkung besteht darin, dass die Orientierung der Hand weder über das MRTK noch über die öffentliche API der HoloLens ausgelesen werden kann. Lediglich die absolute Positionsänderung der Hand kann erfasst werden. Zu sehen in Abbildung 2.5 ist ein Auszug aus dem Inputmodul des MRTK für Unity3D.

Das Inputmodul nutzt ein Eventsystem, dass bei Nutzereingaben die Schnittstellen fokussierter Objekte benachrichtigt. Um eine Eingabequelle in Verbindung mit dem MRTK verwenden zu können, muss sie die `IInputSource`-Schnittstelle implementieren. Innerhalb der Schnittstelle sind Events definiert, die durch die Eingabe ausgelöst werden können. Alle aktiven Eingabegeräte müssen beim `InputManager` registriert sein, der dann die Events der Eingabequellen empfangen kann [21].

Die an virtuelle Objekte gekoppelten Klassen, die Events von Eingabequellen empfangen sollen, müssen dazu die entsprechenden Schnittstellen implementieren

[8] engl. *application programming interface*

(vgl. Abbildung 2.5). Die `ISourceStateHandler`-Schnittstelle wird zum Empfang von `SourceDetected`- und `SourceLost`-Events implementiert. Diese Events werden durch die Gestensteuerung ausgelöst, wenn eine Hand in der Ready-Pose innerhalb des Sichtfeldes registriert wird bzw. dieses verlässt. Mithilfe der `IInputHandler`-Schnittstelle lassen sich `SourceDown`- und `SourceUp`-Events verarbeiten. Diese Events entsprechen dem Beugen bzw. Aufrichten des Zeigefingers in der Air-Tap-Geste. Mit den Schnittstellen `IHoldHandler` und `IManipulationHandler` lässt sich die Tap-and-Hold-Geste verarbeiten. Ein `IManipulationHandler` informiert dabei mit `ManipulationUpdated`-Events kontinuierlich über Veränderungen der Handposition während einer aktiven Tap-and-Hold-Geste. Jede Bewegung der Hand kann so beispielsweise auf die Bewegung eines virtuellen Objekts abgebildet werden. Ein `IHoldHandler` dagegen kann lediglich Events auslösen, die über den Beginn und das Ende einer Tap-and-Hold-Geste informieren.

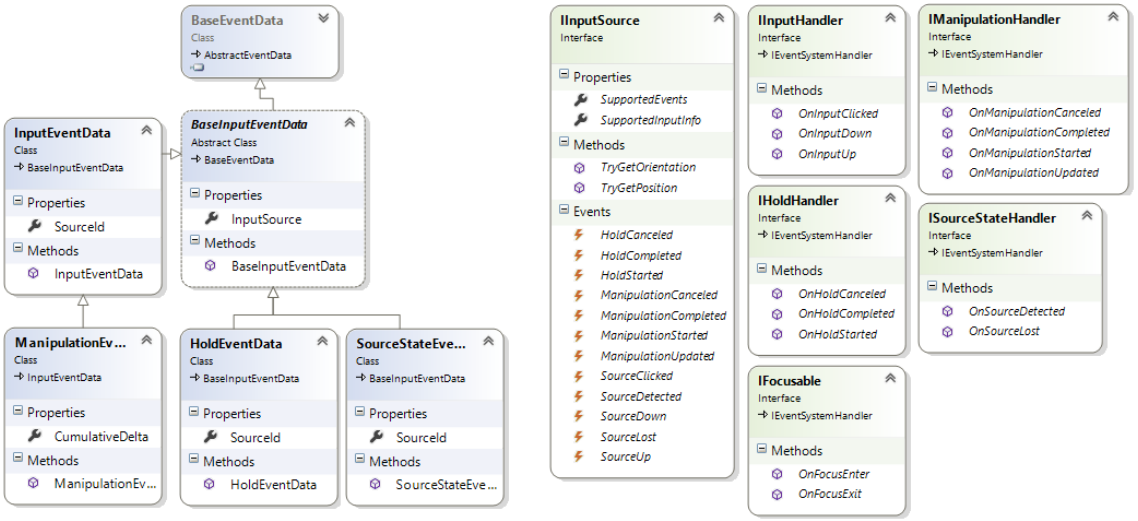


Abbildung 2.5: Auszug aus dem Inputmodul des MRTK für Unity3D [21].

Von Eingabequellen ausgelöste Events werden vom `InputManager` standardmäßig an das aktuell fokussierte Objekt weitergeleitet, sofern die mit dem Objekt verbundene Klasse die dafür notwendige Schnittstelle implementiert. Um Events zu verarbeiten, bevor sie die Objektschnittstelle erreichen, können am `InputManager` weitere globale und modale Empfänger registriert werden. An jedem Empfänger kann entschieden werden, ob das Event verarbeitet oder an nachfolgende Empfänger weitergereicht wird. Events werden in der folgenden Reihenfolge an die Empfänger durchgereicht:

1. Globale Empfänger

Jedes auftretende Event wird stets an alle globalen Empfänger weitergeleitet. Dies geschieht immer unabhängig davon, ob ein Objekt fokussiert ist oder nicht.

2. Modale Empfänger

Abhängig vom Systemzustand oder einer anderen, bestimmten Bedingung. Ein Objekt kann sich beispielsweise für die Dauer einer Manipulation als modaler

Empfänger registrieren, um sicherzustellen, dass `ManipulationUpdated`-Events empfangen werden können, auch wenn es nicht durch die Gaze fokussiert wird.

3. Fokussiertes Objekt

Ein Event das weder durch globale und modale Empfänger verarbeitet wurde, wird an das aktuell fokussierte Objekt weitergeleitet.

4. Fallback-Empfänger

Wird ein Event auch durch ein fokussiertes Objekt nicht verarbeitet (bspw. weil kein Objekt fokussiert wird) können weitere Fallback-Empfänger registriert werden. Dies kann beispielsweise nützlich sein, um dem Anwender Feedback darüber zu geben, warum eine Eingabe fehlgeschlagen ist.

Sind mehrere globale, modale oder Fallback-Empfänger beim `InputManager` registriert, werden die Events nach dem LIFO (Last In, First Out) Prinzip durchgereicht. Der zuletzt registrierte Empfänger erhält das Event dementsprechend zuerst.

2.2.2 Spatial Mapping

Das *Spatial Mapping*[9] [22] ist ein zentrales Element bei der Vermischung von realer und virtueller Umgebung auf der HoloLens. Hierbei wird die ToF-Kamera der HoloLens genutzt, um die Oberflächen des Raumes durch ein dreidimensionales Polygonnetz zu rekonstruieren (siehe Abbildung 2.6 und 2.7). Die Vermessung des Raumes erfolgt in Echtzeit. Die Tiefenkamera misst dabei kontinuierlich die Distanzen zwischen den Raumoberflächen und der Brille. Anhand der sich ändernden Datenbasis wird die virtuelle Repräsentation des Raumes laufend aktualisiert. So können auch Änderungen in der physischen Umwelt, beispielsweise durch verschobene Umgebungsobjekte, stetig auf das virtuelle Abbild des Raumes übertragen werden.

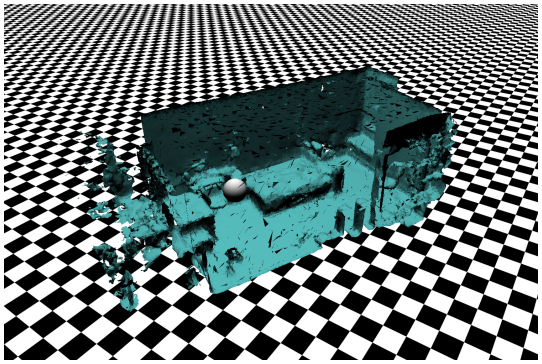


Abbildung 2.6: Raumrekonstruktion bestehend aus 72.640 Polygonen.

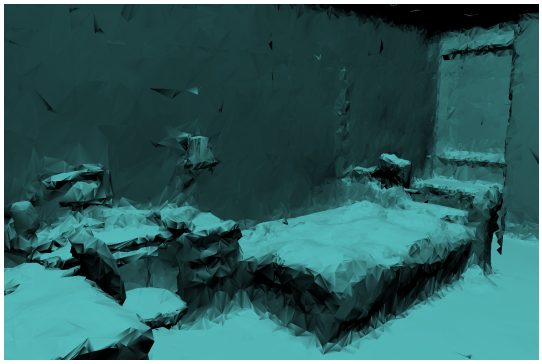


Abbildung 2.7: Virtueller Raum aus Sicht des Anwenders.

Die Positionierung virtueller Objekte im physischen Raum stellt eine Hauptanwendung des Spatial Mappings dar. Virtuelle Objekte können auf den Oberflächen

[9] dt. etwa „Räumliche Abbildung“

des virtuellen Raumes, der deckungsgleich mit dem physischen Raum ist, abgestellt werden oder präzise an die Wände des Raums geheftet werden. Des Weiteren hilft der virtuell rekonstruierte Raum dabei, realistische Physiksimulationen umzusetzen. Eine virtuelle Kugel, die auf den Oberflächen des Polygonnetzes entlang rollt und vom Tisch auf den Boden des Raumes fällt, trägt dazu bei, den Eindruck von Immersion zu verstärken. Nicht zuletzt wird durch das Spatial Mapping die Verdeckung virtueller Objekte durch physische Objekte aus der realen Umwelt ermöglicht. Das Rendering virtueller Objekte wird dabei davon abhängig gemacht, ob es Sichtlinien zu den virtuellen Objekten gibt, die nicht durch die Oberflächen des Polygonnetzes unterbrochen werden.

2.3 Virtual Satellite

Virtual Satellite ist ein MBSE-Tool zur Konzeption von Raumfahrzeugen in frühen Entwicklungsphasen [23]. Durch die virtuelle Modellierung und Simulation von Raumfahrzeugen in Virtual Satellite kann die Durchführbarkeit eines Projekts besser beurteilt werden. Konflikte zwischen einzelnen Raumfahrzeug-Komponenten werden dabei bereits früh identifiziert und wechselseitige Abhängigkeiten aufgelöst. Das in Virtual Satellite verwendete, systemübergreifende Datenmodell (vgl. Abschnitt 1.1) ermöglicht den Aufbau einer hierarchisch geordneten Komponentenstruktur bestehend aus Systemen, Subsystemen, Elementen etc. Jede Komponente kann mit Parametern versehen werden, wie beispielsweise Masse oder Größe. Die Experten arbeiten an lokalen Kopien des Datenmodells und ein Rollenmanagement-System bestimmt, welchen Experten die Befugnis zur Modifikation bestimmter Systemkomponenten zugeteilt wird. Mit Apache Subversion (SVN) ist ein System zur Versionsverwaltung implementiert. So können die einzelnen Experten ihre Ergebnisse miteinander synchronisieren und Änderungen an der Systemkonfiguration nachverfolgen.

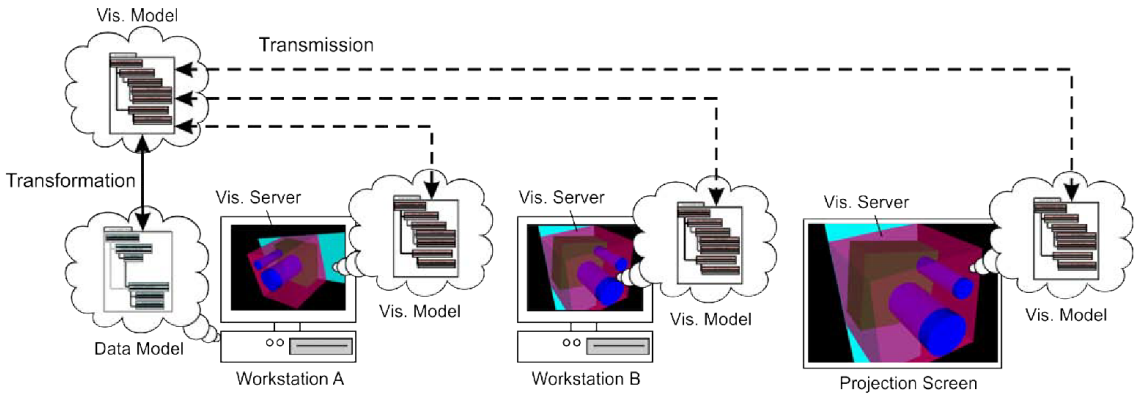


Abbildung 2.8: Transformation des Datenmodells und anschließende Verteilung des Visualisierungsmodells an alle verbundenen Visualisierungsclients [2].

Wie in Abschnitt 1.1 dargelegt, wurde Virtual Satellite um eine verteilte Visualisierungskomponente erweitert. Das Datenmodell wurde hierzu mit zusätzlichen Parametern für Konfigurationsdaten angereichert. Diese Daten beinhalten beispielsweise die geometrische Form und Farbe bestimmter Systemkomponenten. Außerdem wurden Parameter für die Position und Orientierung einer Komponente relativ zu seiner Elternkomponente hinzugefügt.

Das Systemmodell wird im ersten Schritt in ein Visualisierungsmodell konvertiert, welches alle für die Visualisierung relevanten Parameter enthält. Dabei ist das Visualisierungsmodell in Form eines Szenegraphen aufgebaut. Bei einem Szenegraphen handelt es sich um eine hierarchisch geordnete Struktur aus mehreren Knoten. Die Knoten können dabei mehrere Kindkomponenten, aber nur eine Elternkomponente haben. Das Modell wird von einer Visualisierungsanwendung, basierend auf dem Visualization Toolkit (VTK), interpretiert und dargestellt. Hierbei werden Änderungen am Systemmodell augenblicklich mit dem Visualisierungsmodell synchronisiert. Jeder Nutzer ist damit in der Lage, die mit dem Versionsverwaltungssystem synchronisierte Kopie des Systemmodells auf einer lokalen Instanz von Virtual Satellite zu visualisieren. Jede Instanz von Virtual Satellite ist gleichzeitig Visualisierungsserver und Visualisierungsclient. Die Visualisierungsclients anderer Instanzen können sich mit dem eigenen, oder einem beliebigen anderen Visualisierungsserver verbinden, jedoch stets nur mit einem zur selben Zeit. Der Server synchronisiert dann die lokalen System- und Visualisierungsmodelle aller Clients. Diese Kommunikation erfolgt bidirektional. Eine Änderung des Modells am Szenegraphen eines Clients wird automatisch an den Server geschickt, welcher es wiederum an die übrigen Clients weiterleitet.

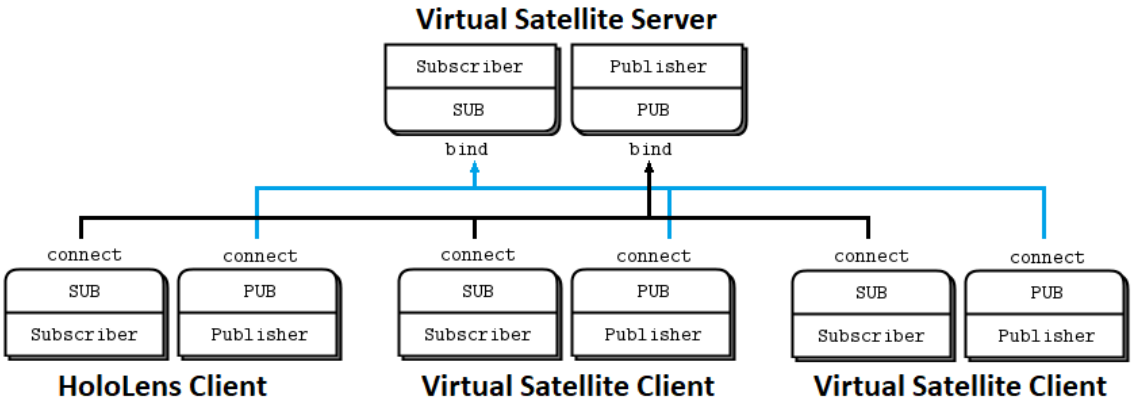


Abbildung 2.9: Messaging-Architektur basierend auf dem Publisher-Subscriber-Muster [24].

Szenegraphen werden vor ihrer Verteilung über das Netzwerk mit Hilfe von Protocol Buffers (protobuf) serialisiert. Bei protobuf handelt es sich um ein sprachneutrales Serialisierungsformat, mit dem sich strukturierte Daten effizient in einen binären Datenstrom umwandeln lassen. Dazu wird eine für diesen Zweck eigens entwickelte Schnittstellen-Beschreibungssprache verwendet. Der so entstandene Datenstrom wird anschließend mit Hilfe von ZeroMQ übertragen. Bei ZeroMQ handelt es sich um eine asynchrone Messaging-Bibliothek für verteilte Anwendungen. Die in ZeroMQ verwendeten Sockets sind portabel, da sie von den zugrundeliegenden Netzwerkprotokollen abstrahieren [24]. Sie erzwingen dabei die Umsetzung bestimmter Entwurfsmuster (z.B. Request-Reply, Publisher-Subscriber) für den Nachrichtenaustausch, wodurch sich softwareseitig schnell Messaging-Topologien umsetzen lassen.

Die Messaging-Architektur von Virtual Satellite basiert auf einem Publisher-Subscriber-Muster. Der Server und alle Clientanwendungen verfügen über jeweils ein Subscriber[10]- und ein Publisher[11]-Socket (vgl. Abbildung 2.9). Über Publisher-

[10] dt. Abonnent
[11] dt. Veröffentlicher

Sockets werden Szenegraphen verschickt und mittels Subscriber-Sockets können diese entgegengenommen werden. Um Szenegraph-Aktualisierungen empfangen zu können, wird das Subscriber-Socket einer Clientanwendung mit dem Publisher-Socket des Servers verbunden. Das Subscriber-Socket des Servers verbindet sich wiederum mit den Publisher-Sockets aller verbundenen Clients. Clientseitige Manipulationen am Szenegraphen werden so an den Server übermittelt, der die Änderungen schließlich an alle verbundenen Clients weiterreicht.

2.4 HoloLens-Visualisierungsclient

Zur Überprüfung der in Abschnitt 1.1 dargelegten Thesen zur Eignung immersiver AR-Anwendungen für die Konzeption von Raumfahrzeugen wurde eine konkrete Implementierung in Form einer HoloLens-Anwendung entwickelt. Die Visualisierungsanwendung soll Wissenschaftler und Ingenieure bei der Entwicklung, Montage und Bewertung von Raumfahrzeugen in der CEF unterstützen. Dies setzt voraus, dass die Anwendung in der Lage ist, von Virtual Satellite generierte Szenegraphen zu empfangen und rekonstruieren. Sie muss dazu in die bestehende Client-Server-Architektur von Virtual Satellite integriert werden. Eine Instanz von Virtual Satellite nimmt hierbei stets die Rolle eines autoritativen Servers gegenüber Clientinstanzen von Virtual Satellite und anderen Clientanwendungen, wie dem Visualisierungsanwendung für die HoloLens, ein.

Aus den vorausgegangenen Betrachtungen ergibt sich (vgl. Abschnitt 2.2 und 2.3) eine Werkzeugkette die zur Realisierung der Clientanwendung beiträgt. Die Nutzung des MRTK setzt die Entwicklung der Anwendung mit Hilfe von Unity3D voraus. Das MRTK verfügt über eine API in der Skriptsprache C-Sharp. Entsprechend wird auch die Programmlogik der Clientanwendung durch eine C-Sharp API realisiert. Zur Einbindung der Anwendung in die Client-Server-Architektur von Virtual Satellite sind daher zusätzlich die .NET Portierungen von ZeroMQ (NetMQ) und protobuf (protobuf-net) notwendig. Hierbei schränkt die UWP Laufzeitumgebung die Implementierungsfreiheit stark ein. NetMQ unterstützt die Entwicklung für die Universal Windows Platform erst seit der aktuellen beta-Version (4.0.0.4-beta). Für protobuf-net ist lediglich eine veraltete Version kompatibel, deren Schnittstellen-Beschreibungssprache auf einer veralteten Syntax basiert. Diese ist mit der von Virtual Satellite verwendeten protobuf-Version nur bedingt kompatibel.

Aufgabe der HoloLens-Anwendung ist zunächst die Entgegennahme des serialisierten Szenegraphen. Dazu verbindet sich die Anwendung mittels eines NetMQ Subscriber-Sockets mit dem Publisher-Socket der Serverinstanz von Virtual Satellite. Dabei wird die Clientanwendung zu Beginn durch einen Drei-Wege-Handschlag als Empfänger authentifiziert. Ist die Registrierung gelungen, werden Änderungen am Szenegraphen über den Server an die HoloLens-Anwendung weitergeleitet. So empfangene Szenegraphen müssen anschließend mit Hilfe von protobuf-net deserialisiert werden. Aus dem deserialisierten Szenegraphen können die Visualisierungsparameter der einzelnen Komponenten schließlich ausgelesen werden. Anhand dieser Parameter wird der Szenegraph in eine Unity3D-Szene überführt und laufend aktualisiert. Um clientseitige Manipulationen am Szenegraphen zur Validierung an den

Server schicken zu können, wird das Publisher-Socket der HoloLens-Anwendung mit dem Subscriber-Socket der Serverinstanz von Virtual Satellite verbunden.

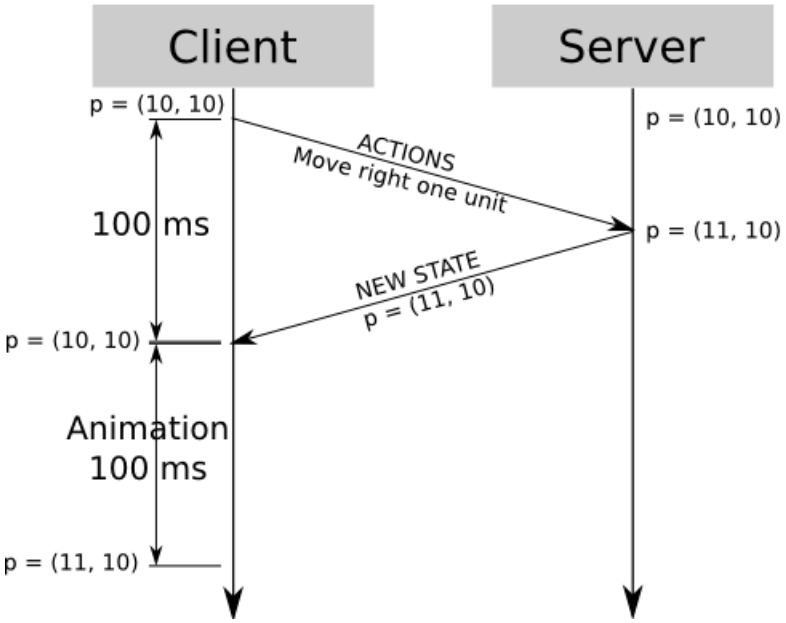


Abbildung 2.10: Latenzen bei Manipulation des lokalen Szenegraphen [25].

Die autoritative Rolle des Servers in der bestehenden Kommunikationsarchitektur dient dem Szenenabgleich aller verbundenen Clients. Ein clientseitig manipulierter Szenegraph wird zur Validierung der Veränderungen an die Serverinstanz von Virtual Satellite gesendet. Der Server schickt den Szenegraphen anschließend an den Sender, sowie an alle übrigen verbundenen Client-Anwendungen, zurück. Erst dann wird die clientseitig durchgeführte Manipulation auf den lokalen Szenegraphen übertragen. Durch die Deserialisierung empfangener Szenegraphen, die Validierung, anschließende Serialisierung und Weiterleitung an alle verbundenen Client-Anwendungen über das Netzwerk entsteht ein Mehraufwand bei der Verarbeitung von Szenegraph-Veränderungen. Manipulationen am lokalen Szenegraphen eines Clients werden dadurch erst nach erheblichen Verzögerungen umgesetzt. Abbildung 2.10 verdeutlicht das Problem anhand der Veränderung der Position eines Objekts mit einer Verzögerung von 100 Millisekunden. Die Veränderung der Position am lokalen Szenegraphen des Clients erfolgt dabei erst nach erfolgter Rücksendung durch den Server.

Mit einer Prädiktion des lokalen Szenegraph-Zustands wird dieses Problem für die HoloLens-Anwendung umgangen [25]. Der Prädiktion liegt die Annahme zugrunde, dass clientseitige Veränderungen des Szenegraphen stets durch den Server validiert und zurückgeschickt werden. Dabei wird eine Manipulation umgehend lokal umgesetzt, ohne auf die Validierung durch den Server zu warten. Ausgesendete Szenegraphen werden in einer Liste gespeichert. Wird ein Szenegraph empfangen, wird dieser mit den gespeicherten Szenegraphen verglichen. Dabei wird geprüft, ob die empfangenen Szenegraphen mit der in der Liste gespeicherten Ereigniskette übereinstimmen. Publisher- und Subscriber-Sockets verfügen intern über eine Warteschlange[12] und verarbeiten Nachrichten nach dem FIFO (First In, First Out) Prinzip.

[12] engl. *Queue*

Daher kann davon ausgegangen werden, dass Nachrichten in der selben Reihenfolge empfangen werden, in der sie verschickt wurden.

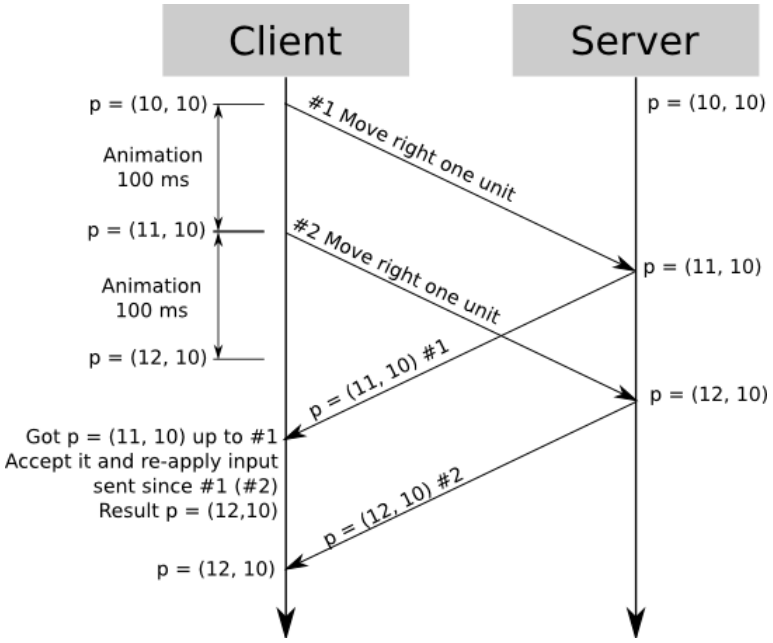


Abbildung 2.11: Client-Prädiktion des Szenegraph-Zustands [25].

Abbildung 2.11 verdeutlicht die Funktionsweise der Prädiktion in einem Beispiel. Die Manipulation der Objektposition wird hier unmittelbar umgesetzt und zur Validierung an den Server übermittelt. Wenn der Client vom Server die Bestätigung erhält, sind die Manipulationen am Client bereits durchgeführt. Es wird nun lediglich geprüft, ob die Transformationen bestätigt wurden.

3 Interaktion

Zur Konzeption eines umfassenden Interaktionsentwurfs ist es zunächst sinnvoll, Teilaspekte der Interaktion in virtuellen Umgebungen zu isolieren und gesondert zu betrachten. Bowman et al. unterscheiden in 3D-Benutzerschnittstellen zwischen vier grundlegenden Aspekten der Interaktion [10].

- **Systemsteuerung:** Hierunter fallen Steuerungsbefehle, die auf Applikationsebene wirksam sind.
- **Selektion:** Die Auswahl eines oder mehrerer virtueller Objekte aus einer Menge an Objekten in der virtuellen Umgebung.
- **Manipulation:** Betrifft die Veränderung der „internen“ Parameter eines virtuellen Objekts, wie dem Gewicht, oder seiner visuellen Repräsentation.
- **Navigation:** Beschreibt Aspekte der motorischen und kognitiven Kontrolle von Position und Orientierung.

In den folgenden Abschnitten wird auf die Teilaspekte der Systemsteuerung, Selektion und Manipulation mit besonderem Bezug auf Augmented Reality eingegangen. Die in der HoloLens-Anwendung implementierten Lösungen werden vorgestellt und begründet. Die Navigation wird in den folgenden Betrachtungen ausgeklammert.

3.1 Spezifikation der Anforderungen

Vor der Entwicklung eines Interaktionskonzepts ist die Analyse des Nutzungskontexts und eine daraus resultierende Spezifikation der Anforderungen notwendig. Die technischen Rahmenbedingungen wurden bereits im Grundlagenteil ausführlich erörtert. Um die Anforderungen an die HoloLens-Anwendung besser vestehen zu können, werden im Folgenden die Rahmenbedingungen der Raumfahrzeugmontage und die Arbeitsprozesse in der CEF genauer untersucht.

Bei der Raumfahrzeugmontage bestehen hohe Anforderungen an die Präzision. Zum Beispiel müssen die Reaktionsräder eines Satelliten im Verhältnis zueinander passgenau angeordnet werden, damit eine Lagerregelung gelingt. Bei der Positionierung von Systemkomponenten muss daher auch eine effektive Feinjustierung möglich

sein. Zudem muss eine simultane Manipulation aller Reaktionsräder möglich sein, um diese im Raumfahrzeug positionieren zu können ohne dabei die Ausrichtung der Reaktionsräder im Verhältnis zueinander durch individuelle Transformationen zu stören. Dies erfordert die Möglichkeit zur gleichzeitigen Selektion und Manipulation mehrerer Systemkomponenten. Satelliten und andere Raumfahrzeuge können des Weiteren große Dimensionen haben. Um einen Satelliten mit mehreren Metern Spannweite in Originalgröße betrachten und manipulieren zu können, ist daher ein ebenso großer Interaktionsraum notwendig. Nicht zuletzt sind Raumfahrzeuge komplexe Systeme, die aus einer Vielzahl verschiedener Komponenten bestehen, die dicht beieinander liegen. Dies kann die Selektion bestimmer Raumfahrzeugkomponenten erschweren, die durch andere verdeckt werden.

Die CEF wurde eingerichtet um die Arbeit in den frühen Planungs- und Designphasen A/0 und B im Produktlebenszyklus eines Raumfahrzeugs zu unterstützen [26]. Der Fokus liegt dabei darauf, das wissenschaftliche Arbeiten und die Kommunikation zwischen Experten unterschiedlicher Domänen zu fördern. Die CEF verfügt über zwölf Plätze (vgl. Abbildung 1.1) für Wissenschaftler und Ingenieure [2]. Die Sitze sind in einem Halbkreis zu einer Wand hin ausgerichtet, an der drei große Bildschirme für Präsentationen angebracht sind. In ein- bis zweiwöchigen Designstudien wird jedem Experten eine Disziplin zugeordnet, wie zum Beispiel Mechanik, AOCS oder oder Antrieb. Nach einer Präsentation der Anforderungen an das Raumfahrzeug wird mit der Arbeit in mehreren Sitzungen begonnen. In jeder Sitzung arbeiten die Ingenieure in kleinen Gruppen an ihren entsprechenden Subsystemen. Nach einer Sitzung, die typischerweise zwei bis drei Stunden dauert, evaluieren die Ingenieure ihre Ergebnisse und diskutieren individuelle Probleme um sich auf die nächste Sitzung vorzubereiten. Dieser Prozess wird iterativ wiederholt, bis ein Konzept entworfen ist, dass die Anforderungen an jedes Subsystem erfüllt. Die so entstandene Konfigurationsskizze wird einem Integrationsingenieur übergeben. Dieser erstellt mit Hilfe geeigneter CAD-Software anhand der Skizze und der Parameter des Systemmodells schließlich ein detailliertes Konstruktionsmodell des Satelliten [2]. Dabei sollte der Integrationsingenieur möglichst wenig Interpretationsspielraum haben, was die Positionierung und Ausrichtung der Komponenten anbelangt. Dies erfordert bereits in der Konfigurationsskizze eine präzise Positionierung der Komponenten.

Aus dem Nutzungskontext und den Anforderungen des Raumfahrzeugbaus ergeben sich die folgenden Kernanforderungen an die Benutzerschnittstelle der HoloLens-Anwendung:

- Selektion verdeckter Objekte.
- Selektion mehrerer Objekte zur simultanen Manipulation aggregierter Komponenten.
- Möglichst hohe Präzision bei der Translation und Rotation der virtuellen Objekte.
- Möglichst großer Interaktionsraum für die Selektion und Manipulation virtueller Objekte.

Bei der Umsetzung der Anforderungen sollte zudem darauf geachtet werden, dass die entwickelten Interaktionsmetaphern möglichst effizient und intuitiv sind.

3.2 Systemsteuerung

Eingaben zur Durchführung von Systemfunktionen, beispielsweise das Speichern eines Dokuments, oder zur Konfiguration des Systemzustands sind ein zentraler Bestandteil der Mensch-Maschine-Interaktion. Bowman et al. [27] definieren die Systemsteuerung als Gesamtheit der Benutzereingaben bzw. Kommandos, welche die folgenden Aufgaben erfüllen:

1. Durchführung einer Systemfunktion.
2. Änderung des Systemzustands.
3. Änderung des Interaktionsmodus.

Bei klassischen Desktopschnittstellen sind entsprechende Systemsteuerungtechniken basierend auf dem WIMP (Window, Icon, Menü, Pointer) Konzept üblich [14]. Für 3D-Benutzerschnittstellen, insbesondere auch für AR-Anwendungen, bestehen jedoch spezifische Anforderungen, die berücksichtigt werden müssen. Sie betreffen Faktoren, welche die Benutzerfreundlichkeit und Gebrauchstauglichkeit der Systemsteuerung beeinflussen können. Gesichtspunkte der Wahrnehmung, Komplexität und Ergonomie stellen hierbei wichtige anwenderbezogene Faktoren dar. Aspekte der Wahrnehmung betreffen in erster Linie die Sichtbarkeit von Systemsteuerungselementen. Speziell in 3D-Anwendungen können diese verdeckt werden. Auf dies ist besonders bei graphischen Eingabeschnittstellen zu achten. Des Weiteren ist es notwendig, dem Anwender Systemzustandsänderungen und den aktuellen Systemzustand transparent zu machen. Die Komplexität der Systemsteuerung hängt im Wesentlichen von ihrem funktionalen Umfang und der strukturellen Tiefe der Steuerungselemente ab. Eine große Zahl an Steuerungselementen, die in mehreren Kategorien bzw. Subkategorien strukturiert sind, kann den kognitiven Anspruch der Systemsteuerung erhöhen [27]. Mit steigender Komplexität wird das Auffinden und Auslösen bestimmter Funktionalitäten zunehmend schwieriger. Hierbei müssen Umfang der Funktionalität und die Komplexität der Systemsteuerung gegeneinander abgewogen werden. Ergonomische Aspekte betreffen Bewegungsabläufe, die mit einem bestimmten Eingabegerät verbunden sind, sowie die Erreichbarkeit von Steuerungselementen. Die Größe und Position von Steuerungselementen spielt dabei eine wichtige Rolle und sollte eine möglichst komfortable Selektion ermöglichen. Die mit einem Eingabegerät verbundenen Bewegungsabläufe stellen ein weiteres wichtiges Kriterium dar. Sollen beispielsweise Drehbewegungen der Hand als Eingabeparameter genutzt werden, muss die begrenzte Auslenkbarkeit der Armgelenke berücksichtigt werden.

Bowman et al. identifizieren Grundtypen von Systemsteuerungstechniken, aus denen sich komplexere Steuerungmetaphern ableiten lassen (vgl. Abbildung 3.1). Diese Techniken umfassen die Steuerung über physische Eingabegeräte, graphische Benutzerschnittstellen sowie die Sprach- und Gestensteuerung [27]. Multimodale Techniken kombinieren unterschiedliche Eingabetechniken, zum Beispiel die Sprachmit der Gestensteuerung. Unter Berücksichtigung der Eingabemodalitäten der HoloLens und der vorausgegangenen Analyse der Anforderungen und des Nutzungskontexts (siehe Abschnitt 3.1) konzentrieren sich die folgenden Betrachtungen vor allem auf Systemsteuerungstechniken, die auf graphischen Schnittstellen, der Gestensteuerung und multimodalen Eingabetechniken basieren.

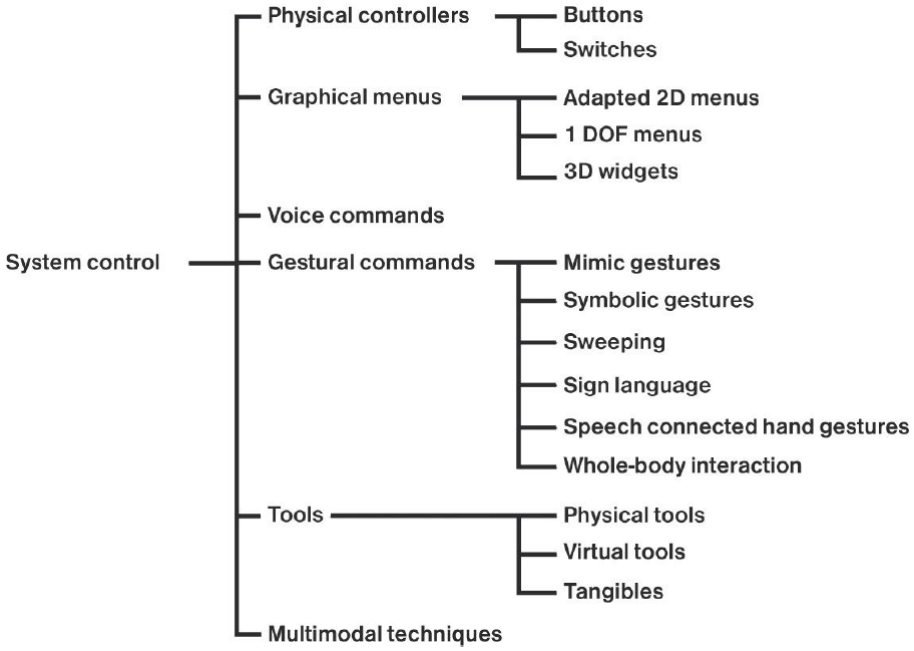


Abbildung 3.1: Taxonomie für Systemsteuerungstechniken [27].

Benutzerschnittstellen, die auf zweidimensionalen graphischen Menüs basieren, werden häufig für Desktop-Anwendungen verwendet. Die weit verbreitete Nutzung klassischer Menüelemente wie Buttons, Toolbars und Pop-Ups macht die Verwendung von dreidimensionalen Entsprechungen dieser Menüelemente zur beliebten Lösung für 3D-Benutzerschnittstellen. Solche Menüelemente eignen sich, um eine Vielzahl an Funktionen zu strukturieren. Insbesondere in 3D Umgebungen besteht jedoch die Gefahr, dass graphische Menüs die virtuelle Umwelt verdecken. Außerdem ist die Selektion der Menüelemente in 3D Umgebungen schwieriger [27]. 3D-Widgets können helfen, die Schwächen einfacher Adaptionen von 2D-Schnittstellen auszugleichen. Durch Widgets wird eine enge Kopplung von Steuerungselement und Funktionalität erreicht [27]. Häufig werden Widgets in der Nähe des Zielobjekts positioniert, auf dem sie agieren sollen. Anstatt beispielsweise den Modus der Transformation (Translation, Rotation oder Skalierung) in einem Menü zu konfigurieren, können entsprechende Widgets in der Nähe des zu manipulierenden Objekts verwendet werden. Die Manipulation der Translation eines Objekts kann dann durch die Manipulation des Translationswidgets erfolgen. Die Bestimmung des Interaktionsmodus und die Manipulation selbst werden so in einer Aktion gebündelt. Widgets sind überdies kontextsensitiv, da sie erst nach der Selektion eines bestimmten Objekts an diesem erscheinen.

Bei graphischen Menüelementen oder Gesten handelt es sich um explizite Techniken der Systemsteuerung. Dabei ruft ein Anwender eine entsprechende Systemfunktion auf, indem er dedizierte Menüelemente anwählt bzw. Gesten durchführt. Es stellt sich in diesem Zusammenhang die Frage, wie sich Änderungen der Systemkonfiguration auch implizit in die Interaktionsabläufe der virtuellen Umgebung einbetten lassen [28]. Ein multimodaler Ansatz hierfür findet sich bei Wilson und Oliver [29]. Dix et al. schlagen eine kontextbasierte Interaktion vor, in der die aktuelle Situation und Umgebung, sowie der aktuelle Systemzustand als zusätzliche Eingabeparameter einer Interaktion dienen [30]. Beispielsweise können bestimmte Funktionalitäten automatisch aktiviert bzw. deaktiviert werden, sobald ein virtuelles

Objekt selektiert oder deselektiert wird. Auf diese Weise kann die Anzahl notwendiger expliziter Steuerungselemente verringert werden. Als Basis zur Realisierung einer kontextbasierten Steuerung können Zustandsautomaten dienen [31].

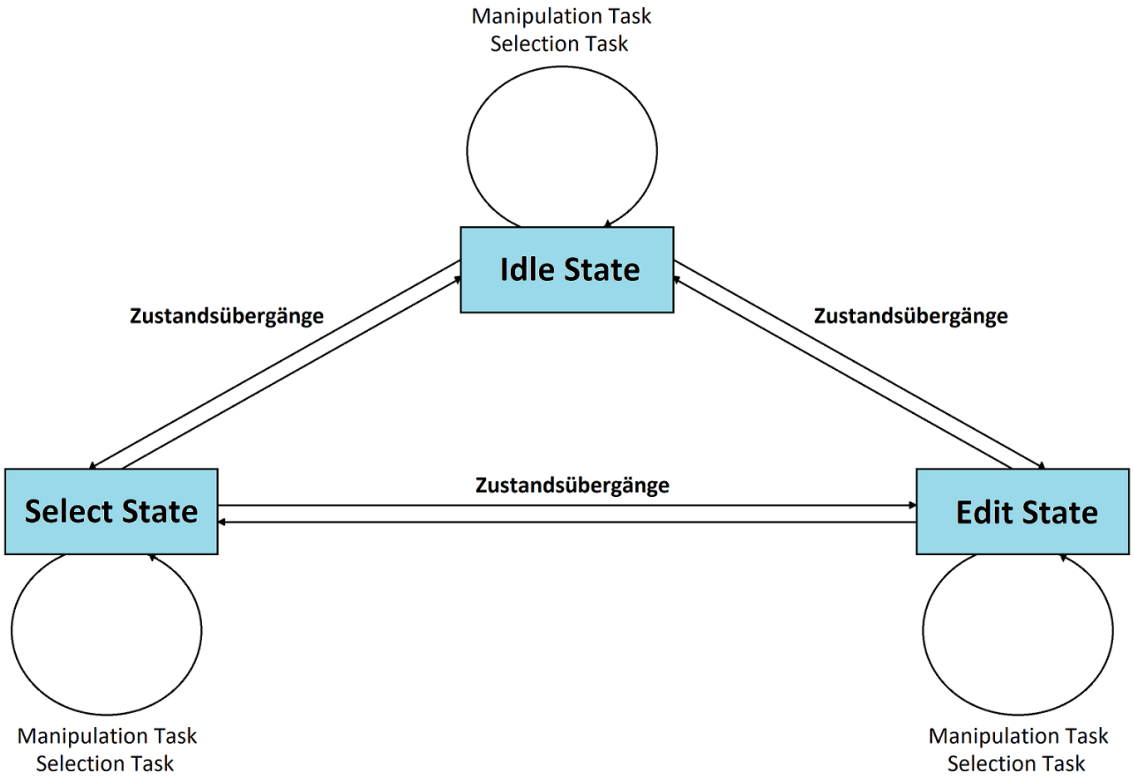


Abbildung 3.2: Zustandsautomat der HoloLens-Anwendung.

In der HoloLens-Anwendung wurden zusätzlich zur Systemsteuerung über graphische Menüs 3D-Wigets sowie Gestensteuerung mit Zustandsautomaten kombiniert. Der Zustandsautomat definiert dabei drei Zustände, in denen sich das System befinden kann (vgl. Abbildung 3.2). Um intuitive Zustandsübergänge und transparente Gesten-Steuerung zu ermöglichen, wird auf die multimodale Eingabetechnik der HoloLens aufgebaut. Multimodale Techniken kombinieren mehrere Eingabequellen zu einer einzelnen, eindeutigen Eingabe. Die genutzten Eingabequellen sind zum einen die Gaze (das aktuell fokussierte Objekt) und die Gestensteuerung. Zusammen mit dem aktuellen Systemzustand und dem aktuellen Zielobjekt wird ein Kommando konstruiert, das eindeutig entweder mit einem Zustandsübergang oder einer Selektions- bzw. Manipulationsaktion korrespondiert.

- **Idle State:** „Neutraler“ Zustand und Ausgangspunkt der Zustandsmaschine.
- **Select State:** In diesem Zustand lassen sich komplexe Selektionsaufgaben wie sie Selektion verdeckter Objekte durchführen (vgl. Abschnitt 3.3).
- **Edit State:** Dieser Zustand dient der Durchführung von Manipulationen an einem Zielobjekt (vgl. Abschnitt 3.4)

Ein Zustand definiert unterschiedliche Sätze an Kommandos, die entweder einem Zustandsübergang oder einer Interaktion entsprechen. Ein Kommando wird durch die Durchführung einer Handgeste ausgelöst. Weitere kontextabhängige Informationen werden dem Kommando als zusätzliche Eingabeparameter hinzugefügt. Dem Kommando kann so eine eindeutige Aktion zugeordnet werden. Jedes Kommando setzt sich aus den in Tabelle 3.1 aufgelisteten Eingabeparametern zusammen. Zu einem ausgelösten Kommando werden die Eingabeparameter in einer Datenstruktur gespeichert und durch den Zustandsautomaten ausgewertet. Bei einer Hold-and-Drag-Geste im Edit State mit Fokus auf ein Skalierungswidget wird beispielsweise eine entsprechende Skalierung des aktuellen Zielobjekts vorgenommen. Die Skalierungslogik ist dabei in einer Klasse implementiert, deren Schnittstelle von der abstrakten Klasse `ManipulationTask` geerbt werden muss. Bei der Manipulation eines Skalierungswidgets wird eine Instanz dieser Klasse erzeugt.

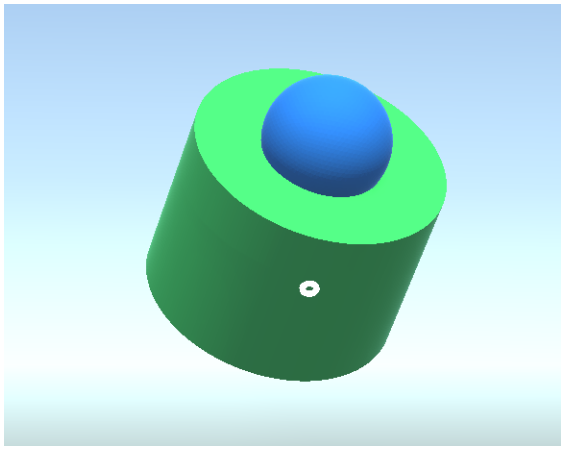


Abbildung 3.3: Ein durch den Gaze-Cursor fokussiertes Objekt.

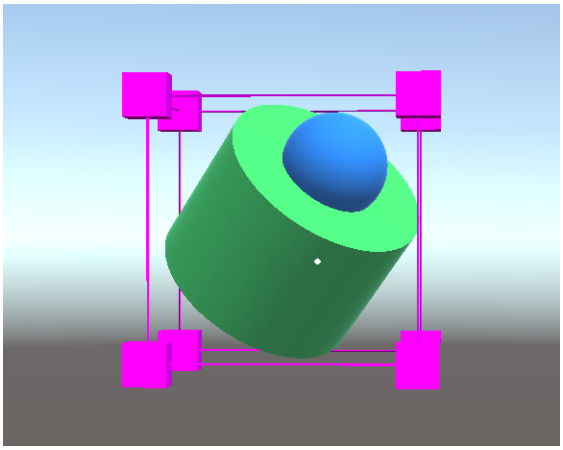


Abbildung 3.4: Selektiertes Objekt mit Bounding Box.

Manipulationen sind Implementierungen der abstrakten `ManipulationTask` Klasse. Die Programmlogik für die Translation und Rotation ist beispielsweise in den Klassen `TargetDragTask` und `TargetRotateTask` implementiert, die beide von der Klasse `ManipulationTask` erben. Sie deklariert die abstrakten Methoden `Start`, `Update` und `End`. Die Zustandsmaschine kann über die einheitliche Schnittstelle unabhängig von der konkret ausgelösten Manipulation die Ausführung dieser steuern. Mit einem `ManipulationTask` können kontinuierliche Manipulationen durchgeführt werden. In der `Start`-Methode können die benötigten Daten zusammengetragen und die initialen Bedingungen festgehalten werden. Beispielsweise kann die Handposition zum Startzeitpunkt der Manipulation zwischengespeichert werden und in folgenden `Update`-Aufrufen als Referenzpunkt für weitere Berechnungen verwendet werden (vgl. Abschnitt 3.4). Die `Update`-Methode dient der kontinuierlichen Aktualisierung der Manipulation. Dies betrifft zum Beispiel die Aktualisierungen der Position und Orientierung eines virtuellen Objekts während einer Handbewegung. In der `End`-Methode kann Programmlogik untergebracht werden, die am Ende einer Manipulation durchgeführt werden muss. Auch diskrete Aktionen wie die Selektion eines virtuellen Objekts werden auf diese Weise umgesetzt. Diskrete Aktionen sind Implementierungen der abstrakten Klasse `SelectionTask`. Sie deklariert die abstrakte Methode `Perform`, in der die Programmlogik eines konkreten `SelectionTask` de-

finiert ist. Ein Beispiel ist die Klasse `GeometrySelectTask`, mit der die virtuellen Objekte selektiert werden können.

Geste	Entspricht einer der grundlegenden Gesten Tap, Double-Tap oder Tap-and-Hold (vgl. Abschnitt 2.2).
Aktueller Zustand	Der gegenwärtige Zustand, in dem sich das Interaktionssystem befindet.
Aktueller Fokus	Entspricht dem durch die Gaze fokussierten Objekt. Ein Objekt gilt als fokussiert, sobald der Gaze-Cursor mit einem Objekt kollidiert (siehe Abbildung 3.3).
Aktuelles Zielobjekt	Das aktuell zur Manipulation ausgewählte Objekt. Das aktuelle Zielobjekt ist erkennbar an einer Bounding Box, welche das Objekt umspannt (siehe Abbildung 3.4).
Einstellungen	Global über die Toolbar konfigurierte Einstellungen.

Tabelle 3.1: Eingabeparameter für ein Kommando des Zustandsautomaten.

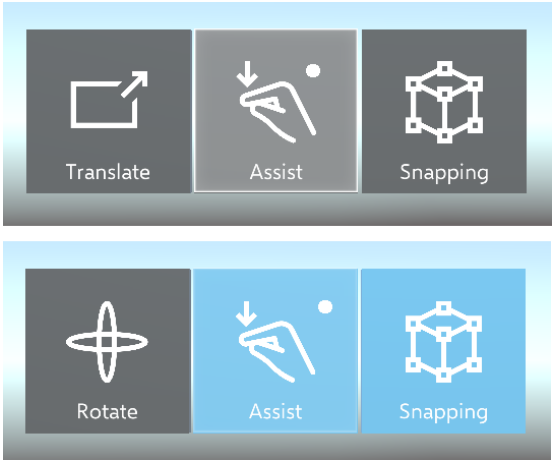


Abbildung 3.5: Toolbar und Toggle-Buttons.

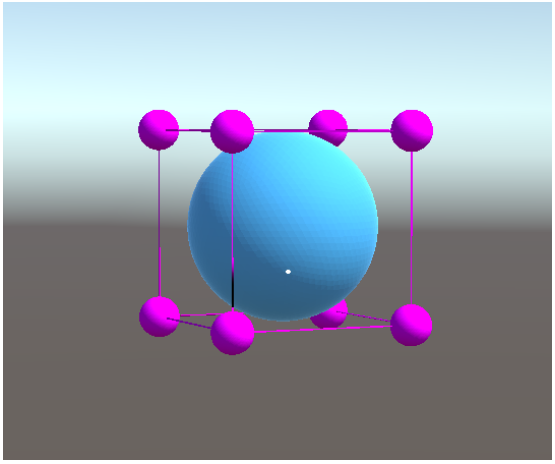


Abbildung 3.6: Bounding Box mit kugelförmigen Widgets.

Weitere globale Konfigurationen, die das Kommando beeinflussen, können an einer Toolbar vorgenommen werden (siehe Abbildung 3.5). Die Toolbar umfasst drei Toggle-Buttons, mit denen der Transformations-Modus (Translation, Rotation), die achsenunabhänge Skalierung oder das Oberflächen-Snapping für Objekte ein- bzw. ausgeschaltet werden können (vgl. Abschnitt 3.4.1 und Abschnitt 3.4.2). Damit die Toolbar stets auffindbar bleibt, folgt sie den Bewegungen des Anwenders und befindet sich immer im Sichtbereich. Um zu verhindern, dass die Toolbar die Sicht auf die Szene behindert, folgt sie den Bewegungen des Anwenders nur bis sie sich im Peripheriebereich des Sichtfeldes befindet. Zusätzlich kann der Transformationsmodus global mit Hilfe einer Double-Tap Geste verändert werden. Dadurch kann der Fokuswechsel auf den Toggle-Button unterbunden werden, der beim Wechseln der Transformationsmodus notwendig wäre. Der Translationsmodus wird hierbei durch würfelförmige Skalierungswidgets (vgl. Abbildung 3.4) gekennzeichnet. Der Rotationsmodus wird hingegen durch kugelförmige Skalierungswidgets gekennzeichnet (vgl. Abbildung 3.6).

3.3 Selektion

Selektion beschreibt die Aufgabe, ein bestimmtes Objekt oder eine Teilmenge an Objekten aus der Gesamtheit aller Objekte in einer virtuellen Umgebung zu identifizieren und auszuwählen [32]. Im Kontext von 3D Benutzerschnittstellen ist dies keine triviale Aufgabe. Im Gegensatz zu klassischen Desktop-Schnittstellen verfügen 3D-Schnittstellen in der Regel über eine größere Anzahl an Freiheitsgraden. Dies macht die Schnittstelle komplexer, weniger leicht verständlich und kontrollierbar [33]. Zusätzlich können Objekte oder UI-Elemente verdeckt werden, was die Objektakquise erschweren kann.

Zur Lösung dieser Probleme lohnt es sich zunächst die Selektion in ihre Teilaspekte zu zerlegen. Bowman et al. [32] identifizieren dabei drei wesentliche Aspekte der Selektion:

- **Indikation:** Die Art und Weise wie angezeigt bzw. identifiziert wird, welches virtuelle Objekt aktuell als zu selektierendes Objekt fokussiert wird.
- **Validierung:** Die Bestätigung der Selektion eines fokussierten Objekts. Entspricht der eigentlichen Auswahl des Objekts.
- **Feedback:** Visuelles, auditives oder haptisches Feedback während der Durchführung der Selektionsaufgabe und über das Ergebnis einer Selektion.

Für die Selektion von Objekten in virtuellen Umgebungen gibt es verschiedene Ansätze. Im Hinblick auf immersive AR- bzw. VR-Geräte wie die HoloLens können im Wesentlichen zwei grundlegende Strategien identifiziert werden. Zum Einen sind dies Ansätze die entweder auf der „Virtual-Hand“ Metapher oder auf der „Ray-Casting“ Metapher basieren [34]. Die Virtual-Hand Metapher baut entweder auf kamerabasierten Trackingverfahren oder auf mit Sensoren augestatteten Eingabegeräten in Form von Handschuhen auf [35]. Die so erfassten Handparameter dienen dann der Nachbildung der Hand in der virtuellen Umgebung. Bei der Ray-Casting Metapher handelt es sich um eine vektorbasierte Technik, bei der ein „Strahl“ in die Szene ausgesendet wird. Dabei wird getestet ob der Strahl mit virtuellen Objekten kollidiert. Dies lässt sich beispielsweise über eine Zeigegeste mit der Hand oder wie im Fall der HoloLens über die Blickrichtung („Gaze“, vgl. Abschnitt 2.2) realisieren.

Da die HoloLens die Positionsparameter der Hand erfassen kann, ist prinzipiell eine Selektion durch die Virtual-Hand Metapher denkbar[1]. Die Ray-Casting Metapher hat gegenüber der Virtual-Hand Metapher bezogen auf die Selektion jedoch entscheidende Vorteile. Von zentraler Bedeutung hierbei ist, dass auch Objekte selektiert werden können, die außerhalb der Reichweite des Anwenders liegen. Die

[1] In immersiven AR-Applikationen ist im Gegensatz zu VR-Applikationen eine visuelle Repräsentation der Hand offensichtlich nicht notwendig. Die virtuelle Repräsentation der Hand umfasst daher in diesem Fall lediglich „interne“ Parameter wie die Handposition im Koordinatensystem der virtuellen Umgebung.

Reichweite des Ray-Casts kann beliebig angepasst werden. Im Fall der HoloLens-Anwendung reicht es nun, weit entfernte Objekte mit dem Gaze-Cursor zu fokussieren. Der Interaktionsraum umfasst daher alle für den Anwender sichtbaren Objekte. Überdies benötigt die Ray-Casting Technik lediglich zwei rotatorische Freiheitsgrade für eine effektive Selektion zu ermöglichen. Ware und Lowther zeigten, dass eine geringere Anzahl an Freiheitsgraden die Selektionseffizienz steigert, da entsprechende Techniken leichter zu verstehen und anzuwenden sind [36].

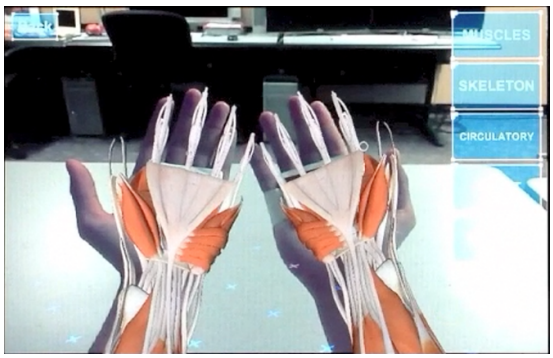


Abbildung 3.7: Offset zwischen realer und virtueller Handposition.

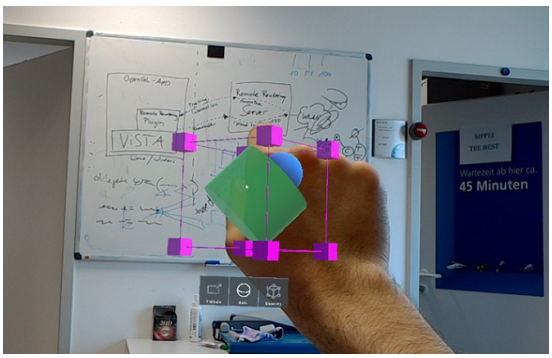


Abbildung 3.8: Hand befindet sich vor virtuellem Objekt, erscheint aber dahinter.

Im Gegensatz zur Ray-Casting Metapher ist bei Verwendung der Virtual-Hand Metapher ein großer Interaktionsraum nicht immanent. Anwender können lediglich Objekte innerhalb der Reichweite einer Armlänge erreichen. Um dieses Problem zu entschärfen, kann die Metapher beispielsweise durch Arm-Extension-Techniken wie GoGo [37] erweitert werden. Diese sind im Kontext von immersiver AR-Technik jedoch kritisch zu sehen. Sie erfordern eine virtuelle, graphische Repräsentation der Hand die sich über die normale Reichweite eines Anwenders hinaus bewegen kann. Derartige Offsets zwischen der Position der realen und der virtuellen Hand können beim Anwender Verwirrung hervorrufen. Zusätzlich muss die Hand Objekte verdecken können, wenn sie sich vor diesen befindet. Abbildung 3.8 zeigt das Problem. Die Hand erscheint hinter dem virtuellen Objekt, obwohl sie sich aus Anwendersicht räumlich vor dem Objekt befindet.

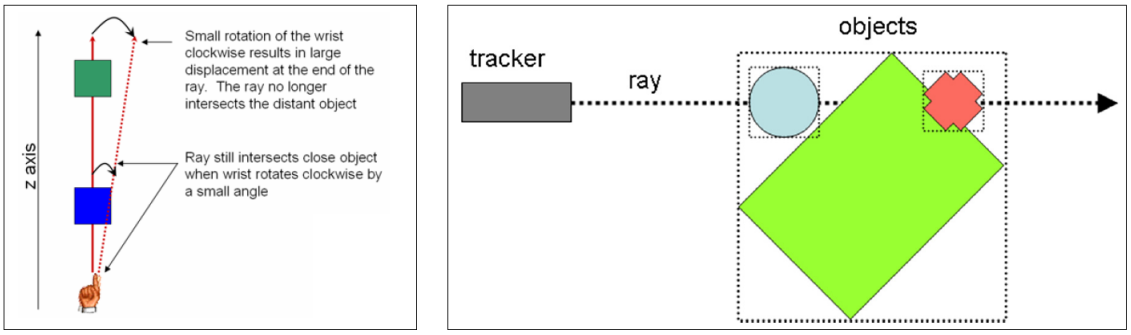


Abbildung 3.9: (a) Rotation eines Ray-Casts [38]. (b) Ray-Casting mit optimierten Bounding Boxen [38].

Die grundlegende Selektionstechnik der HoloLens-Anwendung basiert daher auf einer blickbasierten Ray-Casting Metapher. Die Indikation des zu selektierenden

Objekts wird durch das fokussieren eines virtuellen Objekts mit dem Gaze-Cursor umgesetzt. Die Validierung der Selektion erfolgt über die Gestensteuerung mit einer Tap-Geste. Das so selektierte Objekt wird dann durch eine magentafarbene Bounding Box umspannt, die dem Anwender visuelles Feedback über die erfolgte Selektion gibt (siehe Abbildung 3.4).

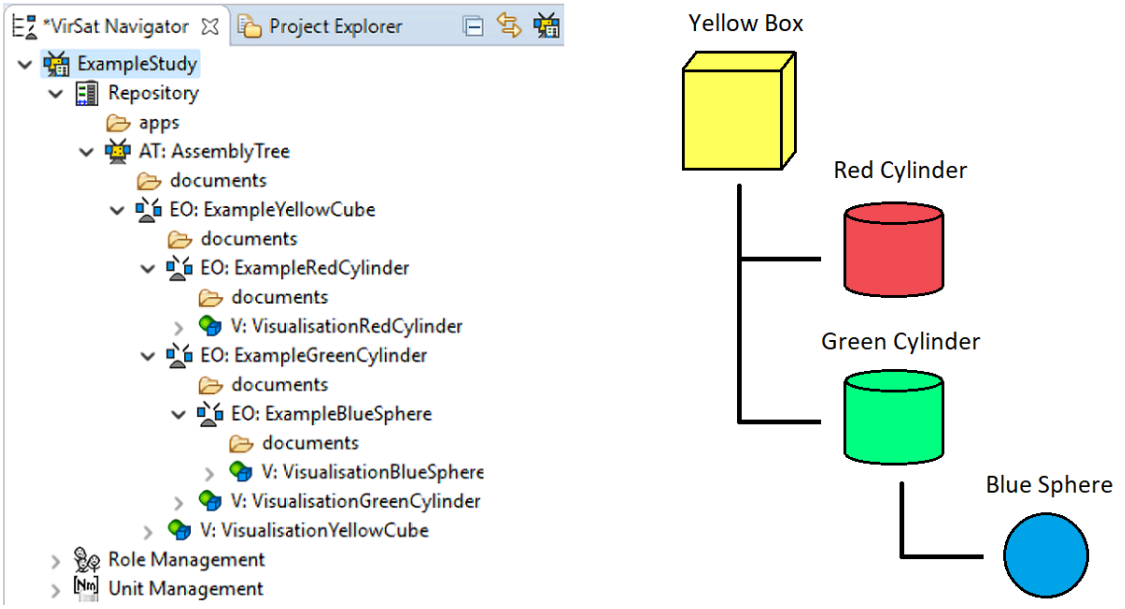


Abbildung 3.10: Beispielhierarchie in Virtual Satellite mit Visualisierungskomponenten.

Problematisch bleibt bei vektorbasierten Techniken wie dem Ray-Casting jedoch die Selektion verdeckter Objekte. In der Regel wird das Objekt mit dem Schnittpunkt selektiert, welcher am nähesten zum Urspung des Ray-Casts liegt. Soll dagegen ein dahinter liegendes Objekt selektiert werden, erfordert dies eine Änderung des Beobachtungspunkts. Dies kann zusätzlich erschwert werden, wenn die Kollisionserkennung eines Objekts durch eine optimierte Bounding Box erfolgt (Abbildung 3.9b). Ein weiteres Problem von Ray-Casting-Ansätzen ist, dass es mit zunehmender Entfernung der virtuellen Objekte schwieriger wird diese zu selektieren. Weit entfernte Objekte nehmen weniger Fläche auf dem Anzeigegerät ein, was bei der Selektion vom Anwender eine höhere Genauigkeit abverlangt. Hinzu kommt, dass kleine Rotationen sich entlang des Strahls vergrößern (Abbildung 3.9a). Bei weit entfernten Objekten können sich bereits minimale Bewegungen der Hand, die unweigerlich auftreten, entlang des Strahls verstärken und zu einem unkontrollierbaren „Zittern“ des Cursors am Ende des Strahls führen. Dies kann die Selektion eines weit entfernten Objekts schwer bis unmöglich machen.

3.3.1 Selektion von Objektgruppen

Neben den dargelegten Problemen kann es bei der Selektion von Objekten zu semantischen Mehrdeutigkeiten kommen. Beispielsweise wenn die Visualisierungen bestimmter Satellitenkomponenten aus mehreren Elementen bestehen. Dies trifft beispielsweise für die Reaktionsräder zu. Es besteht ein Unterschied zwischen der Auswahl eines einzelnen Reaktionsrades und der Auswahl der Satellitenkomponente welche alle Reaktionsräder beinhaltet. Die Selektion ist hier ambivalent, da nicht

immer klar ist, ob der Anwender die einzelne Komponente (ein Reaktionsrad) oder die Elternkomponente (alle Reaktionsräder bzw. Lagerregelungs-Komponente) selektieren will. Gleichwohl ist sowohl die Selektion einzelner Komponenten, als auch hierarchisch übergeordneter Elternkomponenten für eine die effektive Manipulation notwendig. Hat man beispielsweise die Reaktionsräder im Verhältnis zueinander optimal ausgerichtet, erfolgt die weitere Positionierung ausschließlich auf der Ebene Elternkomponente, in der sich die Reaktionsräder einheitlich transformieren lassen.

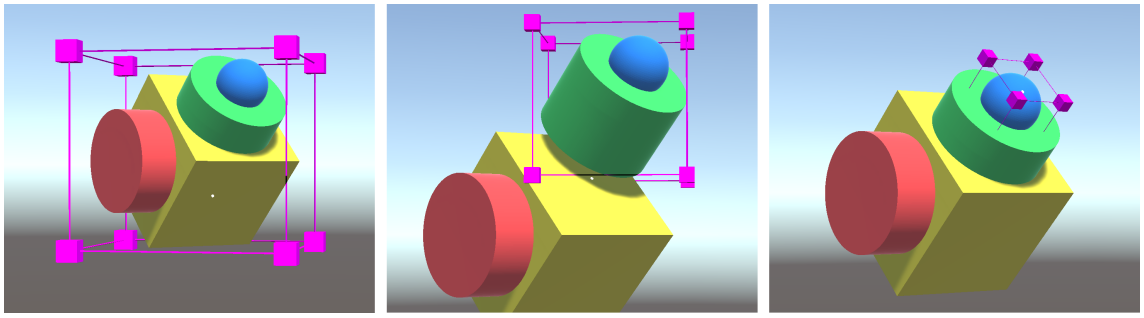


Abbildung 3.11: Objektselektion auf unterschiedlichen Hierarchieebenen.

Um Mehrdeutigkeiten aufzulösen ist in einem ersten Schritt sicherzustellen, dass die Objektselektion auch in der virtuellen Umgebung die Hierarchie des Szenengraphen respektiert. Das in Virtual Satellite visualiserte Systemmodel liegt in einer hierarchisch geordneten Struktur vor (vgl. Abbildung 3.10). Diese Komponentenstruktur dient auch dazu hierarchische Abhängigkeiten wie die zwischen dem gesamten Lagerregelungssystem und den einzelnen Reaktionsrädern, die es beinhaltet, abzubilden. Die Visualisierungsclient-Anwendung baut den Szenengraphen der Visualisierung daher entsprechend der Komponentenstruktur in Virtual Satellite auf. Eine Selektion mehrerer semantisch zusammengehöriger Kindkomponenten kann nun durch Selektion der entsprechenden Elternkomponente erfolgen. Eine magentafarbene Bounding Box schließt in diesem Fall sowohl die Elternkomponente als auch alle Kindkomponenten ein um dem Anwender Feedback über die Hierarchieebene zu geben, auf der die Selektion erfolgt ist (vgl. Abbildung 3.11).

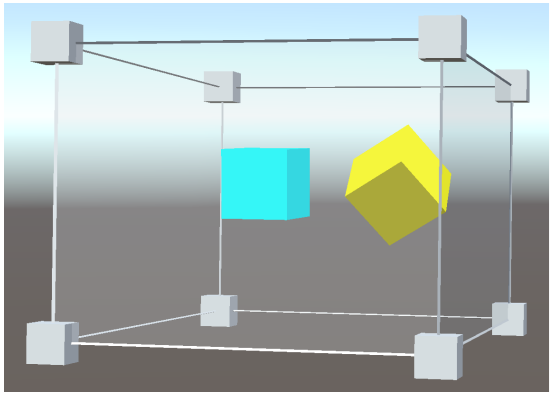


Abbildung 3.12: Selektion mehrerer Objekte durch eine Selektionsbox.

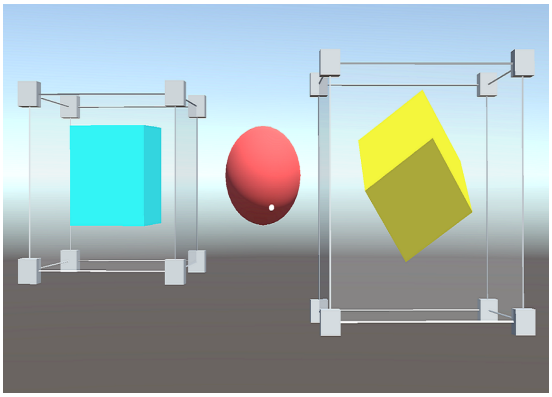


Abbildung 3.13: Objektselektion durch mehrere Selektionsboxen.

Problematisch in der hierarchischen Komponentenstruktur des Systemmodells von Virtual Satellite ist, dass nicht jede Komponente zwangsläufig eine Visualisierung haben muss, jedoch über Kindkomponenten mit Visualisierungen verfügen

kann. In der HoloLens-Anwendung steht dann ebenfalls keine Visualisierung zur Verfügung die durch den Gaze-Cursor zur Selektion anvisiert werden kann. Für solche Fälle, und für Fälle in denen die zu selektierenden Elemente des Szenengraphen nicht in einer direkten Hierarchiebeziehung zueinander stehen, ist eine weitere Selektionsmetapher erforderlich.

Um Objekte in den geschilderten Fällen selektieren zu können, können im Select State durch eine Tap-and-Hold-Geste Selektionsboxen aufgespannt werden (siehe Abbildung 3.12). Die von der Selektionsbox umschlossenen Objekte werden durch eine Tap-Geste als Selektion validiert. Dabei können auch mehrere Objekte punktuell durch mehrere Selektionsboxen selektiert werden (siehe Abbildung 3.13).

3.3.2 Volumenbasierte Selektion

In komplexen Visualisierungen werden Verdeckungen schnell zum Problem für die Objektselektion. Raumfahrzeuge bestehen aus vielen Komponenten, die zum Teil dich beieinander liegen. Objekte können dabei durch andere verdeckt werden oder nur schwer zugänglich und fokussierbar sein. Dies erfordert eine Metapher, die es ermöglicht, auch verdeckte Objekte zu selektieren.

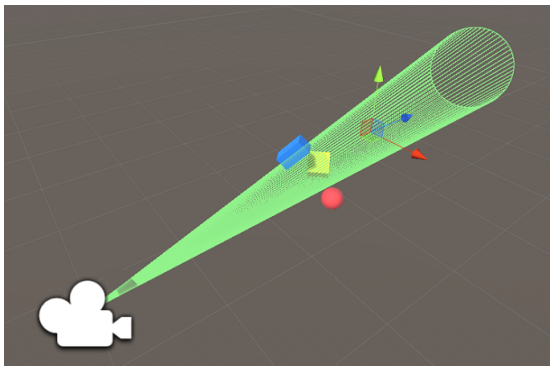


Abbildung 3.14: Cone-Cast zur Eweiterung der Ray-Casting Metapher.

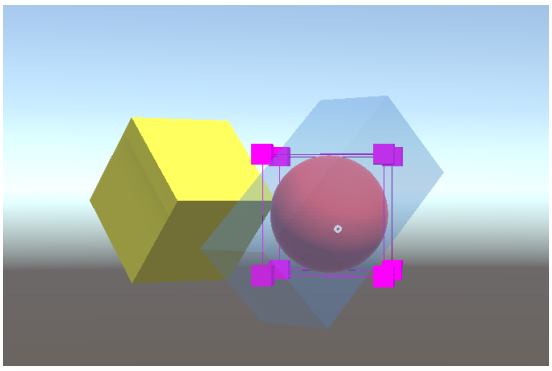


Abbildung 3.15: Selektion einer roten Kugel hinter einem blauen Quader.

Das Problem der verdeckten Selektion für 3D-Benutzerschnittstellen wird in der wissenschaftlichen Literatur viel diskutiert. Grossman und Balakrishnan schlagen eine „Depth Ray“ vor [39]. Hierbei wird der Ray-Cast um einen Tiefenmarker erweitert, der entlang des Strahls verschoben werden kann. Vanacken et al. setzen auf multimodale Techniken [40].

Für die HoloLens-Anwendung wurde eine Cone-Casting Technik implementiert (siehe Abbildung 3.14). Hierbei wird statt einem Strahl ein Kegel für die Kollisionserkennung verwendet [41]. Der Cone-Cast folgt wie der Ray-Cast dem Blick und dient in der Implementierung als Erweiterung der Ray-Casting Metapher. Mit Hilfe des Cone-Casts können die virtuellen Objekte ermittelt werden, die sich in unmittelbarer Nähe des Ray-Casts befinden. So wird bestimmt, welche Objekte das aktuell fokussierte Objekt potentiell verdecken. Die Kollisionserkennung des Cone-Casts ist dabei nur aktiv, wenn sich das System im Select-State befindet. Dies spart Ressourcen, die im Edit-State für die Kollisionserkennung des Snapping-Mechanismus gebraucht werden (vgl. Abschnitt 3.4.2). Durch eine Select-State-Instanz werden zwei

Listen verwaltet. Die erste Liste enthält alle Objekte, die aktuell mit dem Cone-Cast kollidieren. Die zweite Liste enthält Objekte, für die eine Kollisionserkennung mit dem Ray-Cast abgestellt ist. Ein solches Objekt wird transparent dargestellt und ist für den Gaze-Cursor „unsichtbar“ (siehe Abbildung 3.15). Mit Hilfe einer Manipulationsgeste wird ein Objekt dieser Liste zugewiesen. Kollidiert ein durch den Ray-Cast ignoriertes, transparentes Objekt nicht mehr mit dem Cone-Cast, wird es aus der Liste der ignorierten Objekte entfernt und ist damit wieder selektierbar. Der Anwender muss daher keine zusätzlichen Gesten oder Manipulationen durchführen.

3.4 Manipulation

Das Kernziel der HoloLens-Anwendung ist die effektive und inuitive Manipulation von Raumfahrzeug-Komponenten. Die Eignung einer Manipulationstechnik ist dabei in direkter Weise an den Anwendungsfall gekoppelt. Daher müssen entsprechende Techniken ausgehend von der Analyse der Anforderungen und des Nutzungskontexts entworfen werden.

Die Implementierung einer 3D-Objektmanipulation erfordert eine konkrete Definition dieser. Bowman et al. definieren die Manipulation als die interaktive Veränderung charakteristischer Objektparameter. Sie umfassen die folgenden Parameter: Position, Größe, Orientierung, Farbe, Form oder physikalischen Eigenschaften [32]. Für die Raumfahrzeugmontage kann der Umfang der relevanten Aufgaben eingeschränkt werden. Sie umfassen lediglich die kanonischen, geometrischen Transformationen der Translation und Rotation über jeweils drei Freiheitsgrade. Des Weiteren bewahrt die Manipulation die grundsätzliche Form der virtuellen Objekte, d.h. lediglich uniforme Skalierungen sind möglich. Für die einzelnen Manipulationsaufgaben können Parameter identifiziert werden, die eine wichtige Rolle bei der Entwicklung einer Manipulationsmetapher spielen [42]. Beispielsweise kann abhängig davon, ob sich ein Objekt innerhalb der Reichweite einer Armlänge befindet, die Verwendung von Arm-Extension-Techniken sinnvoll sein. Tabelle 3.2 zeigt die relevanten Parameter der Manipulationsaufgaben.

Translation	Distanz und Richtung zur Ausgangsposition des Objekts, Distanz und Richtung zur Zielposition des Objekts, notwendige Präzision der Positionierung
Rotation	Distanz zum Objekt, Ausgangsrotation, Zielrotation, notwendige Präzision der Rotation
Skalierung	Distanz zum Objekt, Ausgangsskalierung, Zielskalierung, notwendige Präzision der Skalierung

Tabelle 3.2: Manipulationsaufgaben und ihre Parameter [32, 42].

Die Eignung einer Manipulationstechnik ist letztlich genau wie die Selektionstechnik an die technischen Voraussetzungen des Eingabesystems gekoppelt (vgl. Abschnitt 2.2.1). Zwei Merkmale haben einen großen Einfluss auf die Anwendbarkeit einer Manipulationstechnik. Zum einen die Anzahl der unabhängig manipulierbaren Freiheitsgrade und zum anderen die Integration bzw. Koordinierbarkeit dieser

Freiheitsgrade [43, 44]. Eingabegeräte, die koordinierte Bewegungen erlauben, ermöglichen die zeitgleiche Aktivierung und Manipulation mehrerer Freiheitsgrade. Bei der freien Positionierung eines Objekts mit der Hand können beispielsweise alle sechs Freiheitsgrade (Translation und Rotation) koordiniert werden. Zhai und Milgram konnten zeigen, dass koordinierte Bewegungsabläufe effizienter sind und weniger physische Ermüdungserscheinungen zur Folge haben [44].

Zhai unterscheidet weiterhin zwischen isomorpher und nicht-isomorpher Manipulation [43]. Isomorphe Manipulationen bilden Bewegungen in der virtuellen Umgebung in strikter Korrespondenz zu Bewegungen des Eingabegeräts ab. Dies entspricht beispielsweise einer 1:1 Abbildung der Handbewegung auf die Bewegung eines virtuellen Objekts. Nicht-isomorphe Manipulationen greifen dagegen auf spezielle Werkzeuge zurück, welche die Effizienz und Gebrauchstauglichkeit der Manipulationstechnik in bestimmten Situationen erhöhen können. Ein Beispiel ist die in Abschnitt 3.3 erwähnte Arm-Extension-Technik GoGo [37]. Der Realismus einer isomorphen 1:1 Abbildung der Handbewegung wird hier zu Gunsten eines größeren Interaktionsraums aufgegeben.

Für die Implementierung der Manipulationsaufgaben müssen die technischen Rahmenbedingungen der HoloLens berücksichtigt werden. Das nur 120×120 Grad Große Sichtfeld der ToF-Kamera schränkt den Interaktionsraum ein. Dies führt dazu, dass die Hand bei Verlassen des Sichtfeldes nicht mehr erfasst werden kann. Der Anwender muss die Hand zurück in das Sichtfeld führen und die Manipulation durch Nachfassen erneut initiieren. Nicht-isomorphe Manipulationstechniken können hier Abhilfe verschaffen, indem sie den Interaktionsraum durch eine konstante Skalierung der Objektbewegung vergrößern. Da ausschließlich die Positionparameter der Hand von den Sensoren der HoloLens erfasst werden, kann eine Eingabe lediglich auf drei Freiheitsgraden gleichzeitig getätigt werden. Dies schränkt die Möglichkeiten zur Durchführung koordinierter Manipulationen ein und macht eine Trennung von Translation, Rotation und Skalierung in separate Manipulationsschritte notwendig. Gleichzeitig ermöglicht dies jedoch die Implementierung einer individuellen Manipulationstechnik für jede der drei kanonischen Operationen.

Translation

Da durch die Tiefenkamera der HoloLens die Position der Hand verfolgt werden kann, ist die Translation virtueller Objekte über die Positionsdifferenz der Hand zwischen zwei Zeitpunkten naheliegend. Um die Position eines Objekts zu manipulieren, wird es selektiert und anschließend im Translationsmodus durch die Tap-and-Hold-Geste ein `TargetDragTask` ausgelöst (vgl. Abschnitt 3.2). Innerhalb der `Update`-Methode wird die Differenz aus zwei aufeinanderfolgenden Handpositionen (Geschwindigkeit der Hand) auf die Objektposition aufaddiert. Zudem wurde eine nicht-isomorphe Abbildung der Handposition auf die Objektposition in einem Verhältnis von 1:3 festgelegt. Die Genauigkeit des Positionstrackings ist jedoch begrenzt. Dies spiegelt sich in groben Objektbewegungen wider. Dem wird durch eine gleitende Mittelung der Handposition aus mehreren zeitlich aufeinanderfolgenden Abtastungen entgegengewirkt. Diese Glättung führt zu einer verzögerten Reaktion

des virtuellen Objekts auf die Beschleunigung oder Verlangsamung der Handbewegung, da mehrere Messungen einen Einfluss auf die gemittelte Handposition haben. Eine nicht wahrnehmbare Verzögerung bei flüssiger Bewegung des virtuellen Objekts ist nach Befund des Verfassers bei einem Abtastumfang von drei bis fünf Messungen gewährleistet.

Skalierung

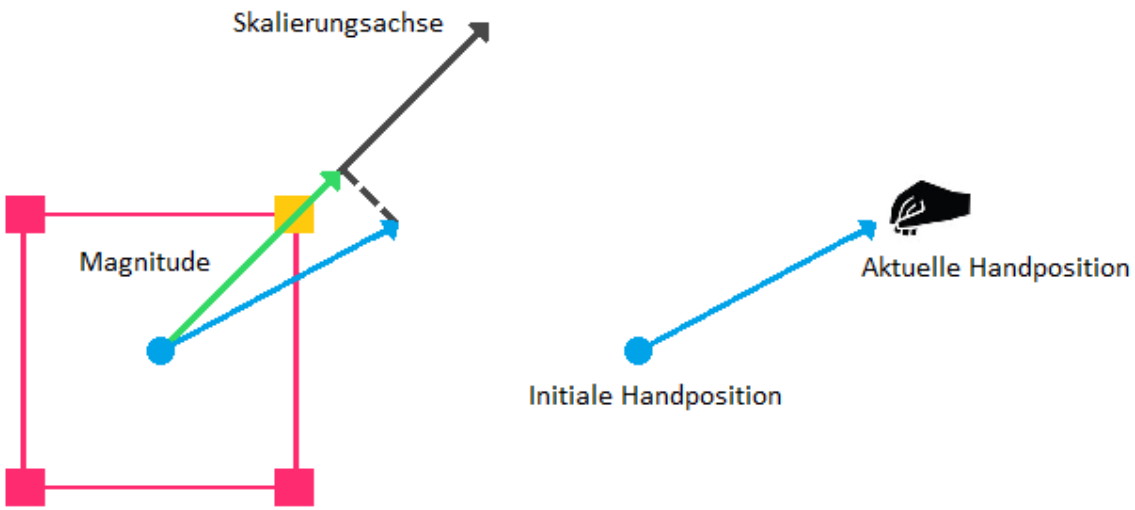


Abbildung 3.16: Projektion der Differenzvektors zwischen initialer und aktueller Handposition auf die Skalierungsachse.

Die uniforme Skalierung ist eine eindimensionale Operation und benötigt nur einen Freiheitsgrad. Sie kann daher auf eine bestimmte Achse beschränkt werden, die dem Freiheitsgrad der Skalierung entspricht. Die Skalierung wurde mit Hilfe von Skalierungswidgets umgesetzt, die an den Ecken der das aktuelle Zielobjekt umspannenden Bounding Box angebracht sind. Durch Fokussierung eines Skalierungswidgets und Durchführung einer Hold-and-Drag-Geste wird ein `TargetScaleTask` ausgelöst. Das aushgewählte Skalierungswidget verändert seine Farbe (vgl. Abbildung 3.17). In der `Start`-Methode wird die aktuelle, absolute Handposition zwischengespeichert. Die Skalierungsachse wird aus Zielobjekt-Mittelpunkt und Widgetposition geformt (vgl. Abbildung 3.16). In der `Update`-Methode wird kontinuierlich der Differenzvektor aus der Initialposition und der aktuellen Position der Hand berechnet und auf die Skalierungsachse projiziert. Die Magnitude der auf die Skalierungsachse projizierten Handbewegung wird schließlich zur Vergrößerung bzw. Verkleinerung des Objekts herangezogen. Die Skalierungswidgets müssen zusammen mit dem Objekt skaliert werden, da sie bei starker Herunterskalierung die Sicht auf das Objekt verdecken können (vgl. Abbildung 3.18). Je stärker die Widgets jedoch herunterskaliert werden, desto schwieriger wird es, sie zu fokussieren. Daher wurde eine minimale Skalierung festgelegt.

Rotation

Ein widgetbasierter Ansatz ist auch für die Implementierung der Rotation denkbar. Die Rotation ist allerdings im Gegensatz zur uniformen Skalierung eine Operation,

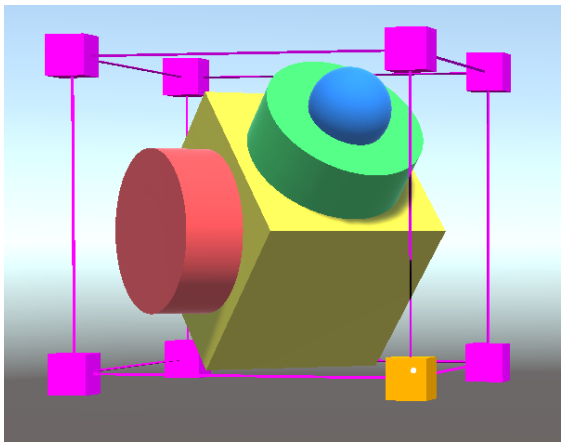


Abbildung 3.17: Manipulation eines Skalierungsiwdgets.

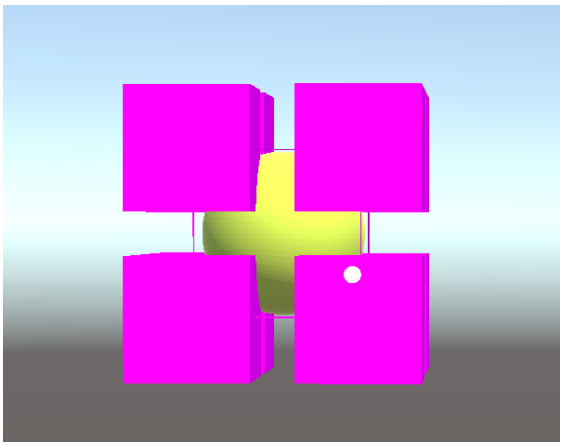


Abbildung 3.18: Skalierungswidgets verdecken die Sicht auf klein skalierte Objekte.

die drei Freiheitsgrade benötigt. Zudem würde das Hinzufügen weiterer Widgets die Selektierbarkeit des Objekts für besonders groß und besonders klein skalierte Objekte weiter erschweren. Ware zeigte, dass die Durchführungszeiten von 3D Rotationsaufgaben im Vergleich zu Translationsaufgaben wesentlich höher sind [45]. Parsons konnte belegen, dass 3D Rotationen kognitiv prinzipiell schwieriger nachzuvollziehen sind als 3D Translationen [46]. In einer Nutzerstudie konnten Frees et al. diese Beobachtung bestätigen [38]. Anwender hatten dabei Schwierigkeiten ein virtuelles Objekt von einer zufälligen Ausgangsrotation in eine Zielrotation zu überführen, da nicht unmittelbar klar war, um welche Achse rotiert werden muss. Durch zufälliges Austesten verschiedener Rotationen mussten die Anwender das Rotationsverhalten des Objekts nachvollziehen. Sobald die Achse jedoch bestimmt wurde, konnte die Rotationsaufgabe zügig gelöst werden. Insbesondere für die Rotation erscheint daher die Möglichkeit der schnellen und effizienten Bedienung mehrerer Freiheitsgrade als sinnvoll. Der Anwender kann so durch eine Erprobung mehrerer Achsen schnell die richtige Rotationsachse finden.

Um eine schnelle, koordinierte Manipulation über mehrere Rotationsachsen zu ermöglichen wurden daher die Positionsdifferenzen[2] der Hand auf die Rotationsachsen abgebildet. Um die Rotation eines Objekts zu manipulieren, wird es selektiert und anschließend im Rotationsmodus durch die Tap-and-Hold-Geste ein `TargetRotateTask` ausgelöst (vgl. Abschnitt 3.2). In der `Start`-Methode wird zunächst das Kamera-Koordinatensystem[3] zwischengespeichert. In der `Update`-Methode wird der Differenzvektor zweier aufeinanderfolgender Handpositionen ermittelt. Der Differenzvektor wird anschließend in das zwischengespeicherte Kamera-Koordinatensystem transformiert. Eine Auslenkung der Hand nach rechts bzw. links aus Anwendersicht entspricht dann der x-Komponente des Differenzvektors, eine Auslenkung nach oben bzw. unten der y-Komponente und eine Auslenkung nach vorne bzw. hinten der z-Komponente (vgl. Abbildung 3.19). Auf diese Weise ist sichergestellt, dass sich die Objektrotation aus jeder Perspektive gleich verhält. Die Magnitude der x-Komponente wird auf eine Rotation des Objekts um die y-Achse des Kamera-

[2] Wie in Abschnitt 2.2.1 dargelegt, kann die Orientierung der Hand mit Hilfe der Tiefenkamera der HoloLens nicht erfasst werden.

[3] Die Kameraposition und -orientierung entspricht der Position und Orientierung der HoloLens im Raum, d.h. der Perspektive des Anwenders.

Koordinatensystems abgebildet. Die Magnitude y-Komponente wird analog dazu auf eine Rotation um die x-Achse und die Magnitude der z-Komponente auf eine Rotation um die z-Achse abgebildet.

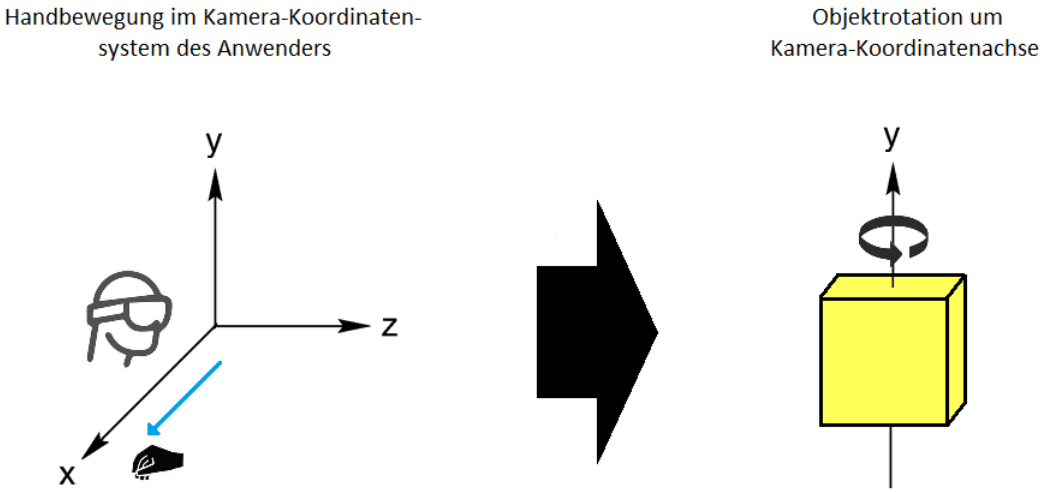


Abbildung 3.19: Abbildung der Handbewegung auf die Objektrotation.

3.4.1 Skalierte Objektmanipulation

Die Anforderungen an die Präzision von Eingaben sind in der Raumfahrzeugmontage ausgesprochen hoch. Die Ausrichtung und Position bestimmter Systemkomponenten können kritische Faktoren sein (vgl. Abschnitt 3.1). Neben der schnellen Positionierung von Systemkomponenten muss daher auch eine effektive Feinjustierung möglich sein. Haptisches und taktiles Feedback, das bei der Manipulation physisch-realer Objekte hilft auch komplexe Aufgaben zu bewältigen, steht bei der Manipulation virtueller Objekte mit der HoloLens nicht zur Verfügung. Darüber hinaus ist die Manipulationsgenauigkeit technischen Einschränkungen unterworfen. Sehr feine Bewegungen können unter Umständen nicht abbildbar sein, wenn der Trackingmechanismus entsprechend kleine Bewegungen nicht registrieren kann.

Es gibt diverse Ansätze zur Verbesserung der Manipulationsgenauigkeit unter Berücksichtigung dieser Einschränkungen. Eine Technik beinhaltet die Skalierung von Benutzereingaben [47, 38, 37]. Während die Techniken von Herder et al. und Poupyrev et al. vor allem der Vergrößerung des Interaktionsraums dienen, unterstützt die PRISM Interaktionstechnik zusätzlich die Erhöhung der der Eingabepräzision durch das Herunterskalieren der Eingabegeschwindigkeit [38]. Da die Eingabepräzision ein wichtiges Kriterium der HoloLens-Anwendung darstellt (siehe Abschnitt 3.1), wird eine Bewegungsskalierung in Anlehnung an die PRISM Interaktionstechnik verwendet.

Der Grundlegende Ansatz besteht darin, die Bewegungsgeschwindigkeit eines virtuellen Objekts in Abhängigkeit von der Bewegungsgeschwindigkeit der Hand

zu skalieren. Das Verhältnis von Eingabegeschwindigkeit[4] und Ausgabegeschwindigkeit[5] wird durch einen Skalierungsfaktor festgelegt. Die mit dem Skalierungsfaktor multiplizierte Eingabegeschwindigkeit entspricht dann der Bewegungsgeschwindigkeit des virtuellen Objekts. Ist der Skalierungsfaktor 1, wird die Bewegungsgeschwindigkeit unskaliert vom Eingabe- auf den Ausgabebereich abgebildet. Die Geschwindigkeit der Handbewegung entspricht der Geschwindigkeit der Objektbewegung. Ein Skalierungsfaktor größer als eins vergrößert die Bewegungsgeschwindigkeit im Ausgabebereich, wohingegen eine Skalierungsfaktor kleiner als eins die Bewegungsgeschwindigkeit verringert. Die Objektbewegung ist in diesen Fällen schneller bzw. langsamer als die Handbewegung.

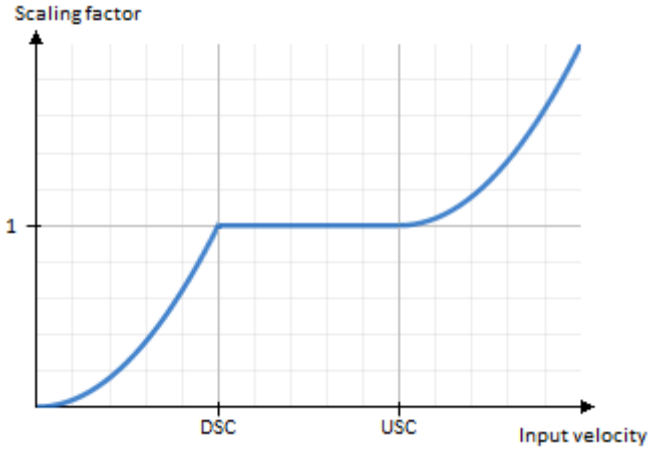


Abbildung 3.20: Bestimmung des Skalierungsfaktors anhand der Eingabegeschwindigkeit. Quadratische Skalierung unterhalb der `DownscalingConstant` (DSC) und oberhalb der `UpscalingConstant` (USC).

Die in der HoloLens-Anwendung implementierte Bewegungsskalierung basiert auf drei Skalierungsmodi: Einen Modus für präzise Objektmanipulation, einen Modus für unskalierte Bewegungen und einen Modus für die schnelle und grobe Objektmanipulation. Vorüberlegungen von Frees et al. [38] und aus Abschnitt 3.2 folgend soll der Wechsel zwischen den Interaktionsmodi idealerweise implizit erfolgen. Ein impliziter Wechsel zwischen den Interaktionsmodi lässt sich unter Bezugnahme auf Fitts‘ Gesetz realisieren [48]. Übertragen auf die Bewegungsgeschwindigkeit besagt Fitts’ Gesetz, dass sich die Geschwindigkeit der Eingabe mit zunehmender Nähe zur Zielposition und Verringerung der Zielgröße (tolerierbare Abweichung von der optimalen Zielposition des virtuellen Objekts) verringert. Konkret bedeutet dies, dass ein Anwender langsame Handbewegungen durchführt, wenn das aktuelle Manipulationsziel eine hohe Präzision erfordert. Umgekehrt werden bei groben Manipulationen schnelle Handbewegungen durchgeführt. Anhand dieser Beobachtung kann die Objektbewegung im Ausgabebereich in Abhängigkeit von der Bewegungsgeschwindigkeit im Eingabebereich skaliert werden. So werden präzise Manipulationen durch verlangsamte Objektbewegungen und grobe Manipulationen durch beschleunigte Objektbewegungen unterstützt.

[4]Geschwindigkeit der Handbewegung
[5]Geschwindigkeit der Objektbewegung

Die Bestimmung des Skalierungsfaktors und der Wechsel zwischen den Interaktionsmodi erfolgt mithilfe von drei Skalierungskonstanten (vgl. Abbildung 3.20). Jede Handbewegung, deren Geschwindigkeit einen kleinen, minimalen Wert unterschreitet wird unterdrückt. Dieser Wert kommt in der Konstanten `MinV` (Minimum Velocity) zum Ausdruck. Bei Handbewegungen, deren Geschwindigkeiten unter der Konstanten liegen, handelt es sich in den meisten Fällen nicht um unbewusste Bewegungen [38]. Der Skalierungsfaktor ist hier null. Befindet sich die Geschwindigkeit der Handbewegung unterhalb einer weiteren Konstanten `DownscalingConstant`, wird die Objektgeschwindigkeit kontinuierlich quadratisch im Verhältnis zur Handgeschwindigkeit herunterskaliert. Der Skalierungsfaktor ist kleiner als eins. Befindet sich die Geschwindigkeit der Handbewegung oberhalb der Konstanten `UpscalingConstant`, wird die Objektgeschwindigkeit kontinuierlich quadratisch im Verhältnis zur Handgeschwindigkeit hochskaliert. Der Skalierungsfaktor ist größer als eins.

Unabhängige Skalierung der Achsen

Die Positionsdifferenz bzw. Handgeschwindigkeit entspricht einem euklidischen Vektor im Welt-Koordinatensystem. Die Projektion der Handgeschwindigkeit auf die Welt-Koordinatenachsen kann dementsprechend leicht aus den einzelnen Vektorkomponenten abgelesen werden. Anstatt die Vektormagnitude zur Errechnung eines Skalierungsfaktors heranzuziehen, kann für jede der Vektorkomponenten ein separater Skalierungsfaktor bestimmt werden. Differenzvektor zunächst vom Welt-Koordinatensystem in das Kamera-Koordinatensystem transformiert. Da sich die Orientierung und Position des Kamera-Koordinatensystems im Welt-Koordinatensystem während der Manipulation kontinuierlich ändert, wird die Kameraorientierung und -position bei der Initiierung einer Manipulationssequenz zwischengespeichert. Während der Manipulationssequenz errechnete Positionsdifferenzen werden dann stets in das zwischengespeicherte Koordinatensystem transformiert. Für jede Komponente der Positionsdifferenz wird ein separater Skalierungsfaktor berechnet. Bewegt der Anwender seine Hand zum Beispiel zügig nach rechts und driftet dabei leicht nach oben, dann wird die x-Komponente des Differenzvektors hoch-, die y-Komponente jedoch herunterskaliert.

Durch die unabhängige Skalierung der Koordinatenachsen können insbesondere Manipulationsaufgaben erleichtert werden, die in koordinierter Form kognitiv schwer nachzuvollziehen sind. Dies ist beispielsweise bei Rotationsaufgaben der Fall [46]. Bewusste Bewegungen entlang einzelner Koordinatenachsen werden unterstützt, da unbewusste minimale Abweichungen in Richtung der anderen Koordinatenachsen unterdrückt werden. Das Kamera-Koordinatensystem stellt hierbei einen sehr starken Referenzrahmen für Anwender dar.

3.4.2 Snapping

Um die Objektpositionierung und -ausrichtung weiter zu erleichtern wurde ein Snapping-Mechanismus implementiert.

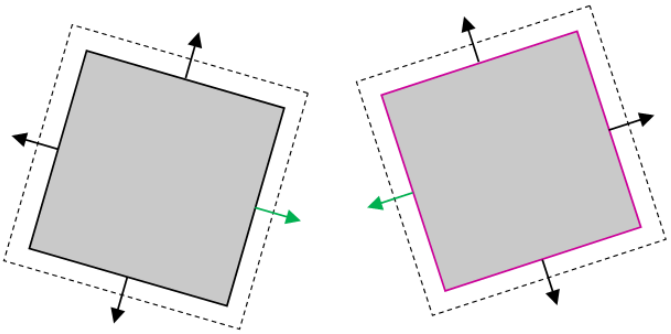


Abbildung 3.21: Objekte rasten nicht ein, da die Hüllkörper nicht kollidieren.

Um das Snapping zu aktivieren, wird auf der Toolbar der „Snapping“-Button getoggelt (siehe Abbildung 3.5). Jedem virtuellen Objekt wird dadurch ein zusätzlicher, unsichtbarer Hüllkörper zugewiesen, der für eine Kollisionserkennung genutzt wird. Dieser Hüllkörper ist um einen kleinen, konstanten Faktor größer, als die Bounding Box, die ein virtuelles Objekt umspannt. Damit zwei Seiten unterschiedlicher Objekte einrasten, müssen zwei Bedingungen erfüllt werden:

- Die Hüllkörper zweier Objekte müssen kollidieren.
- Die Flächennormalen einer Seite müssen ein negatives Skalarprodukt, unterhalb eines Schwellwerts haben (`SnappingConstant`).

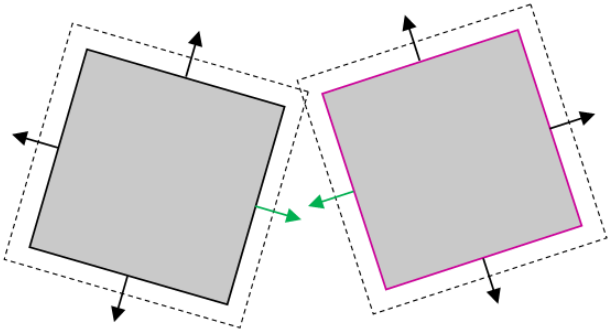


Abbildung 3.22: Objekte rasten ein, da die Hüllkörper kollidieren.

Die Abbildungen 3.21 und 3.22 demonstrieren den Snapping-Mechanismus in einem Beispiel. In Abbildung 3.21 ergibt das Skalarprodukt der grün markierten Flächennormalen einen Wert unter `SnappingConstant`. Die gestrichelt markierten Hüllkörper der Objekte kollidieren jedoch nicht. Daher rasten die Seiten der Objekte nicht ein. In Abbildung 3.21 kollidieren die Hüllkörper dagegen. Das manipulierte Objekt rastet mit der Seite der grün markierten Flächennormalen an der Oberfläche der grün markierten Flächennormalen des Objekts ein, mit dem es kollidiert ist.

4 Evaluation

Die für die HoloLens-Anwendung implementierte Benutzerschnittstelle wird in einer Nutzerstudie mit der Eingabeschnittstelle von Virtual Satellite verglichen. Ziel des Vergleichs ist es, Vor- und Nachteile immersiver AR-Anwendungen für die Raumfahrzeugmontage gegenüber der desktopbasierten Virtual Satellite-Schnittstelle zu ergründen. Anhand von abhängigen Messgrößen quantitativer und qualitativer Art sollen zudem Verbesserungspotentiale für die Weiterentwicklung der HoloLens-Anwendung offengelegt werden. Innerhalb eines iterativen Entwicklungsprozess können die gewonnenen Erkenntnisse in zukünftigen Iterationen der HoloLens-Schnittstelle integriert werden. Die Probanden haben in der Nutzerstudie mit der HoloLens- und der Virtual Satellite-Schnittstelle jeweils drei Docking-Tasks durchgeführt. Dabei wurde sowohl die Bearbeitungszeit der Tasks, als auch die Positions- und Rotationsabweichungen zur Zielausrichtung des zu manipulierenden Objektes gemessen. Zu jeder Schnittstelle füllten die teilnehmenden Personen im Anschluss einen Fragebogen aus (siehe Anhang 1). Der Studienaufbau und die Auswertung orientiert sich vorwiegend an den Darstellungen aus dem Kapitel „Nutzertests“ des Buches „Virtual und Augmented Reality (VR/AR): Grundlagen und Methoden der Virtuellen und Augmentierten Realität“ [49].

4.1 Studienaufbau

An der Nutzerstudie haben insgesamt zwölf Personen teilgenommen. Bei der überwiegenden Mehrheit der Studienteilnehmenden handelte es sich um technisch-affine Personen, die mindestens über grundlegende Kenntnisse im Umgang mit desktopbasierten Computerschnittstellen und in den meisten Fällen auch Erfahrung im Umgang mit 3D-Grafikanwendungen (z.B. Blender, Unity3D) haben. Einige der Teilnehmenden haben grundlegende Erfahrung im Umgang mit der HoloLens.

Aufgrund dieser Voraussetzungen erscheint eine Within-Group-Studie als angemessen. Durch die kleine Anzahl der Studienteilnehmenden würde der Stichprobenumfang in einer Between-Group-Studie für jede der getesteten Schnittstellen sehr klein ausfallen. Außerdem können starke Varianzen in den grundsätzlichen Fertigkeiten im Umgang mit Computerschnittstellen das Ergebnis in einer Between-Group-Studie stark verzerren. Dadurch, dass alle Teilnehmenden beim Within-Group-Design sowohl die Tests an der Virtual Satellite- als auch der Visualisierungsclient-Schnittstelle durchführen, können derartige, subjektbezogene Varianzen herausgefiltert werden.

Ein wesentlicher Nachteil von Within-Group-Studien liegt darin, dass Übungs- und Ermüdungseffekte die Leistung der Teilnehmenden im Verlauf der Messungen beeinflussen können. Um dem entgegenzuwirken, wurde die Testreihenfolge durch Verwürfelung für jede teilnehmende Person ausbalanciert (vgl. Tabelle 4.1).

Nr.	Schnittstelle	Task 1	Task 2	Task 3
1	HoloLens	A	B	C
2	HoloLens	B	A	C
3	HoloLens	B	C	A
4	HoloLens	A	C	B
5	HoloLens	C	A	B
6	HoloLens	C	B	A
7	Virtual Satellite	A	B	C
8	Virtual Satellite	B	A	C
9	Virtual Satellite	B	C	A
10	Virtual Satellite	A	C	B
11	Virtual Satellite	C	A	B
12	Virtual Satellite	C	B	A

Tabelle 4.1: Randomisierung der Testreihenfolge für zwölf Personen.

Für die HoloLens-Schnittstelle wurden alle Alignment-Tools (Snapping und unabhängige Skalierung der Achsen) ausgestellt. Lediglich die Bewegungsskalierung war aktiviert. Jeder Task wurde einmal mit der HoloLens- und einmal mit der Virtual Satellite-Schnittstelle durchgeführt. Die ersten sechs Probanden führten alle Tasks gemäß der Reihenfolge in Tabelle 4.1 zuerst mit der HoloLens- und anschließend mit der Virtual Satellite-Schnittstelle durch. Die nächsten sechs Probanden führten die Tasks zuerst mit der Virtual Satellite- und anschließend mit der HoloLens-Schnittstelle durch. Vor Durchführung der drei Tasks hatten die Probanden die Gelegenheit den Umgang mit der jeweiligen Schnittstelle zehn Minuten zu üben. Vor jedem Task wurde den Probanden ein Video mit Instruktionen gezeigt. In dem Video wurde den Probanden eine Ausgangskonfiguration und eine Zielkonfiguration gezeigt. In den Videos erklärt eine Person den Probanden mit Hilfe von Handgesten die Anforderungen der Zielkonfiguration.

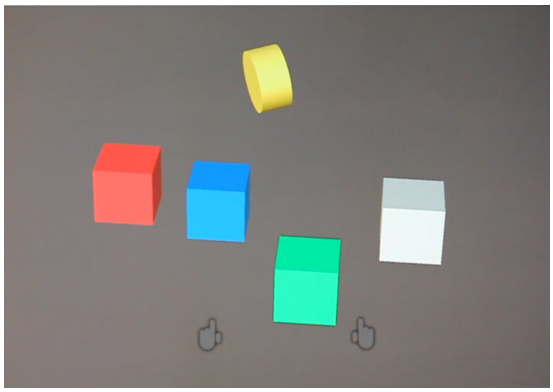


Abbildung 4.1: Ausgangskonfiguration für Task A.

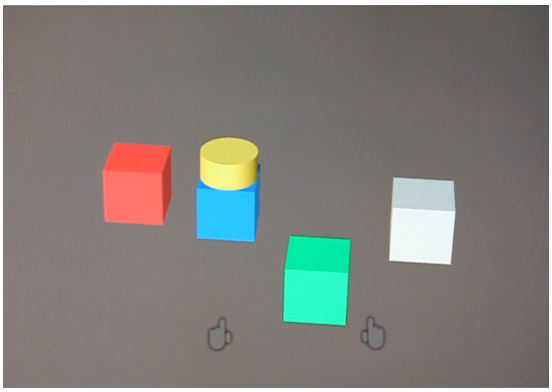


Abbildung 4.2: Zielkonfiguration für Task A.

Im ersten Task (Task A) wurden die Probanden aufgefordert, einen gelben Zylinder präzise auf einem blauen Würfel zu platzieren (vgl. Abbildung 4.1 und 4.2).

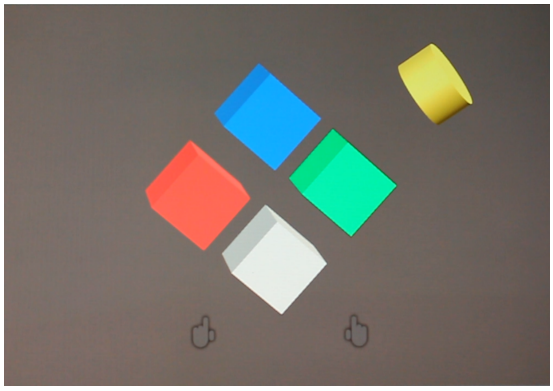


Abbildung 4.3: Ausgangskonfiguration für Task B.

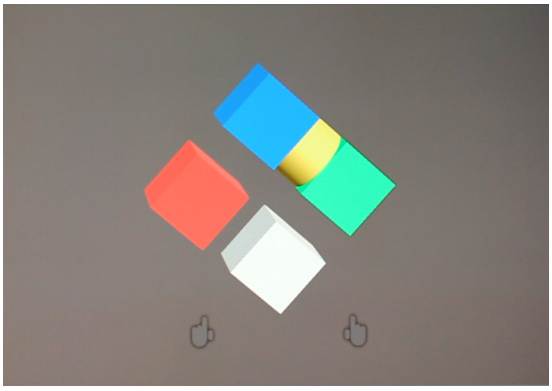


Abbildung 4.4: Zielkonfiguration für Task B.

Im zweiten Task (Task B) wurden die Probanden aufgefordert, einen gelben Zylinder präzise zwischen einem grünen und einem blauen Würfel zu platzieren. (vgl. Abbildung 4.3 und 4.4).

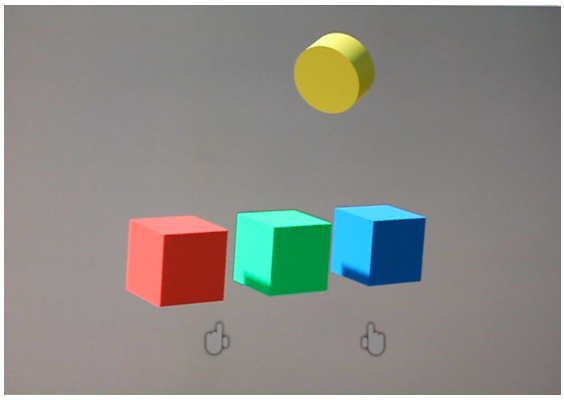


Abbildung 4.5: Ausgangskonfiguration für Task C.

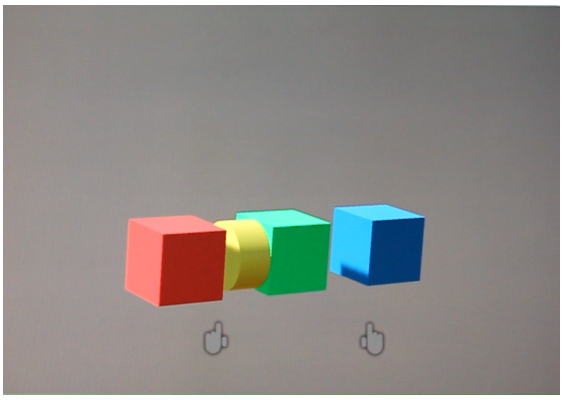


Abbildung 4.6: Zielkonfiguration für Task C.

Im dritten Task (Task C) wurden die Probanden aufgefordert, einen gelben Zylinder präzise zwischen einem grünen und einem roten Würfel, an die Seite des grünen Würfels zu platzieren. (vgl. Abbildung 4.5 und 4.6).

Im Anschluss an die Durchführung der Tasks mit einer Schnittstelle wurde den Probanden ein Fragebogen ausgehändigt. Der Fragebogen ist an den NASA Task Load Index angelehnt. Die Fragen zielten darauf ab, den subjektiv wahrgenommenen Aufwand während der Durchführung der Tasks zu erfassen. Es wurden die folgenden Fragen gestellt:

- How mentally demanding was it to perform the tasks?
- How physically demanding was it to perform the tasks?
- How successful were you in accomplishing what you were asked to do?
- How successful were you in accomplishing the tasks quickly?

- How successful were you in accomplishing the tasks with precision?
- How hard did you have to work to accomplish your level of performance?
- How insecure, discouraged, irritated, stressed and annoyed were you?

Die Probanden konnten zu jeder Frage zwischen sieben Abstufungen wählen und unter jeder Frage einen Kommentar hinzufügen.

4.2 Auswertung

Anhand eines Kolmogorow-Smirnov-Tests kann wird geprüft, ob die erhobenen Messwerte für die Bearbeitungszeit als auch für die Positions- bzw. Rotationsabweichungen normalverteilt sind [49]. Für ein Signifikanzniveau von $\alpha = 0,05$ und $n = 36$ Messwerte (alle Tasks) bzw. $n = 12$ Messwerte (ein Task) ergeben sich kritische Werte für die Ablehnung der Nullhypothese (Messwerte sind normalverteilt) von $0,226$ (Approximation nach Gleichung 4.1) bzw. $0,375$. Für Werte $n \leq 35$ können die kritischen Werte des Kolmogorow-Smirnov-Tests aus einer Tabelle abgelesen werden [49].

$$\frac{\sqrt{-0.5 \cdot \ln \frac{\alpha}{2}}}{\sqrt{n}} \quad für\ n > 35 \tag{4.1}$$

Tabelle 4.2 zeigt die entsprechenden Teststatistiken für die einzelnen Messwerte der Bearbeitungszeit. Die Teststatistiken für die Bearbeitungszeit für beide Schnittstellen unterschreiten in keinem Fall den kritischen Wert von $0,226$ bzw. $0,375$. Die Nullhypothese des Kolmogorow-Smirnov-Tests (Messwerte sind normalverteilt) wird in diesen Fällen daher nicht verworfen und es liegen für die Bearbeitungszeit rational normalverteilte Messwerte vor. Da ein Within-Group-Design gewählt wurde liegt zudem eine verbundene Stichprobe vor. Zur Auwertung der Messwerte für die Bearbeitungszeit wird daher der Paired t-Test verwendet [49]. Es wird die folgende Nullhypothese H_0 und Alternativhypothese H_A formuliert:

H_0 : Der Unterschied zwischen den Bearbeitungszeiten der Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch nicht signifikant.

H_A : Der Unterschied zwischen den Bearbeitungszeiten der Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch signifikant.

Für die Paired t-Tests wird ein Signifikanzniveau von $\alpha = 0,05$ festgelegt. Aus der Alternativhypothese H_A geht zudem bereits hervor, dass ein zweiseitiger Test durchgeführt wird.

Aus Abbildung 4.8 kann entnommen werden, dass die mittlere Bearbeitungszeit für alle Tasks bei der HoloLens-Schnittstelle mit 139,6 Sekunden um 47,6 Sekunden geringer als die Bearbeitungszeit für die Tasks mit der Virtual Satellite-Schnittstelle mit 187,2 Sekunden. Abbildung 4.7 zeigt Boxplots der erhobenen Messdaten. Der Median der Bearbeitungszeit mit der HoloLens-Schnittstelle bei 99,5 Sekunden. Die 25%- und 75%-Quartile liegen bei 75,5 und 193 Sekunden. Die 10%- und 90%-Quartile liegen bei 39 und 294 Sekunden. Der Median der Berabeitungszeit mit der

Schnittstelle	Messwert	Kritischer Wert	Teststatistik
HoloLens	Zeit Gesamt	0,226 ($n = 36$)	0, 1807
HoloLens	Zeit Task A	0,375 ($n = 12$)	0, 1764
HoloLens	Zeit Task B	0,375 ($n = 12$)	0, 1571
HoloLens	Zeit Task C	0,375 ($n = 12$)	0, 2000
Virtual Satellite	Zeit Gesamt	0,226 ($n = 36$)	0, 1838
Virtual Satellite	Zeit Task A	0,375 ($n = 12$)	0, 1261
Virtual Satellite	Zeit Task B	0,375 ($n = 12$)	0, 1210
Virtual Satellite	Zeit Task C	0,375 ($n = 12$)	0, 2299

Tabelle 4.2: Teststatistiken für die in der Nutzerstudie erhobenen Messwerte der Bearbeitungszeit.

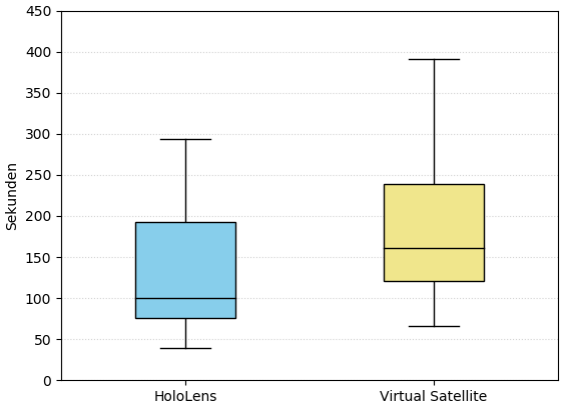


Abbildung 4.7: Boxplots der Bearbeitungszeit aller Tasks: HoloLens (blau) und Virtual Satellite (gelb).

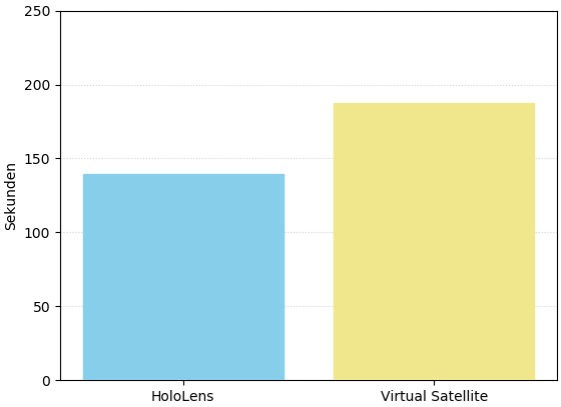


Abbildung 4.8: Arithmetisches Mittel der Bearbeitungszeit aller Tasks: HoloLens (blau) und Virtual Satellite (gelb).

Virtual Satellite-Schnittstelle liegt bei 161 Sekunden. Die 25%- und 75%-Quartile liegen bei 121,25 und 239 Sekunden. Die 10%- und 90%-Quartile liegen bei 66 und 391 Sekunden.

Der Paired t-Test bestätigt die Signifikanz des augenfälligen deutlichen Unterschieds von 47,6 Sekunden in der Bearbeitungszeit. Für die Differenz der Messwerte in der Bearbeitungszeit zwischen der HoloLens- und der Virtual Satellite-Schnittstelle ergibt sich ein p-Wert von $0,0181$. Dieser Wert liegt unter dem geforderten Signifikanzniveau $\alpha = 0,05$, womit die Nullhypothese H_0 verworfen werden kann.

Um sich ein differenzierteres Bild von diesem Ergebnis machen zu können, werden die Messwerte im Folgenden für jeden Docking-Task separat ausgewertet. Die sich stellende Frage hierbei ist, ob es Unterschiede bei der Bearbeitungszeit zwischen den einzelnen Tasks gibt und welche Ursachen diese Unterschiede haben könnten.

Das Balkendiagramm in Abbildung 4.10 zeigt, dass die mittlere Bearbeitungszeit für Task A für die HoloLens-Schnittstelle 118,75 Sekunden und für die Virtual Satellite-Schnittstelle 217,75 Sekunden beträgt. Der Median der Bearbeitungszeit liegt mit der HoloLens-Schnittstelle bei 109 Sekunden und mit der Virtual Satellite-Schnittstelle bei 192,5 Sekunden (vgl. Abbildung 4.9). Die 25%- und 75%-Quartile liegen für die Messungen mit der HoloLens-Schnittstelle liegen bei 85 und 126,75

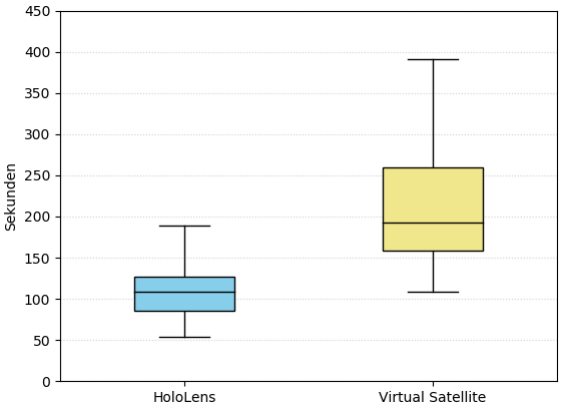


Abbildung 4.9: Boxplots der Bearbeitungszeit für Task A: HoloLens (blau) und Virtual Satellite (gelb).

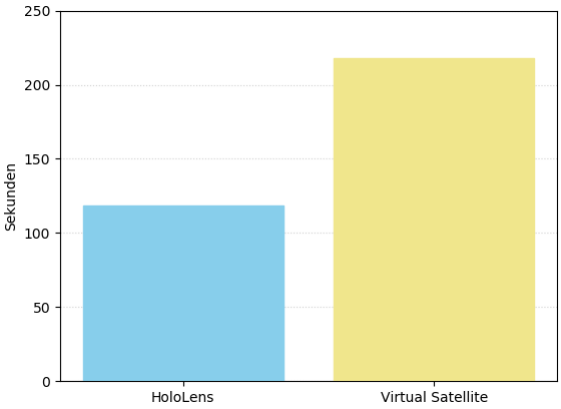


Abbildung 4.10: Arithmetisches Mittel der Bearbeitungszeit für Task A: HoloLens (blau) und Virtual Satellite (gelb).

Sekunden. Die 10%- und 90%-Quartile liegen bei 54 und 189 Sekunden. Für die Messungen mit der Virtual Satellite-Schnittstelle liegen die 25%- und 75%-Quartile bei 158,75 und 259,5 Sekunden. Die 10%- und 90%-Quartile liegen bei 109 und 391 Sekunden. Der Paired t-Test bestätigt die statistische Signifikanz des Unterschieds zwischen den Mittelwerten der Bearbeitungszeit auf beiden Schnittstellen. Es ergibt sich ein p-Wert von $0,0002$. Dieser Wert liegt unter dem geforderten Signifikanzniveau $\alpha = 0,05$ und damit wird die Nullhypothese H_0 verworfen.

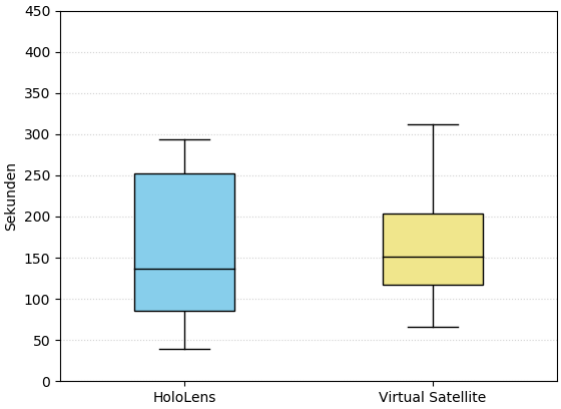


Abbildung 4.11: Boxplots der Bearbeitungszeit für Task B: HoloLens (blau) und Virtual Satellite (gelb).

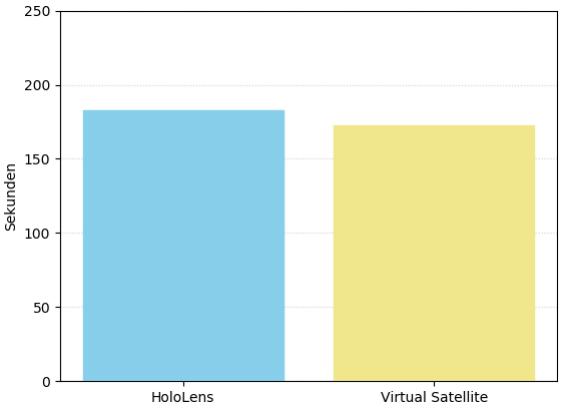


Abbildung 4.12: Arithmetisches Mittel der Bearbeitungszeit für Task B: HoloLens (blau) und Virtual Satellite (gelb).

Das arithmetische Mittel der Bearbeitungszeit liegt für Task B bei 182,58 Sekunden (HoloLens-Schnittstelle) bzw. 172,83 Sekunden (Virtual Satellite-Schnittstelle), zu sehen in Abbildung 4.12. Der Median liegt für die Messwerte der HoloLens-Schnittstelle bei 136,5 Sekunden (vgl. Abbildung 4.11). Die 25%- und 75%-Quartile liegen bei 85 bzw. 251,75 Sekunden. Die 10%- und 90%-Quartile liegen bei 39 und 294 Sekunden. Für die Messwerte der Virtual Satellite-Schnittstelle liegt er Median bei 151,5 Sekunden. Die 25%- und 75%-Quartile liegen bei 117,25 bzw. 203 Sekunden. Die 10%- und 90%-Quartile liegen bei 66 und 312 Sekunden. Die Studienteilnehmer haben Task B mit der HoloLens-Schnittstelle Mittel 9,75 Sekunden langsamer durchgeführt. Ein Paired t-Test zeigt, dass diese Differenz statistisch nicht signifikant ist. Es ergibt sich ein p-Wert von $0,7778$. Dieser Wert liegt über dem geforderten Signifikanzniveau $\alpha = 0,05$, sprich die Nullhypothese H_0 kann nicht verworfen werden.

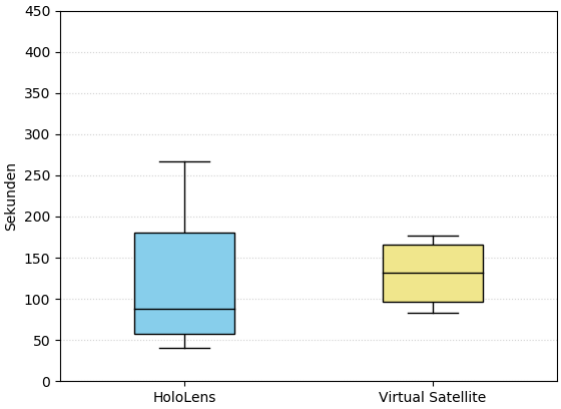


Abbildung 4.13: Boxplots der Bearbeitungszeit für Task C: HoloLens (blau) und Virtual Satellite (gelb).

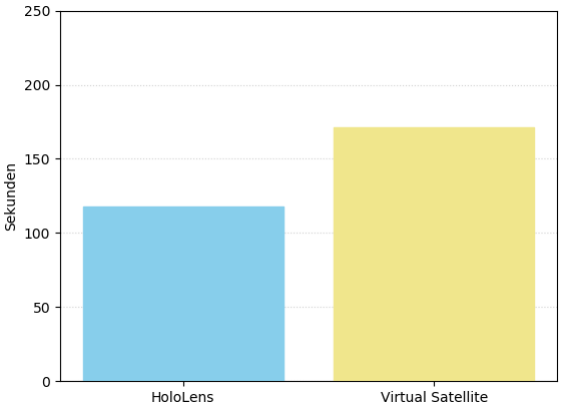


Abbildung 4.14: Arithmetisches Mittel der Bearbeitungszeit für Task C: HoloLens (blau) und Virtual Satellite (gelb).

Zu sehen in Abbildung 4.13 sind die Mittelwerte der Bearbeitungszeit in Höhe von 117,58 und 171 Sekunden der HoloLens- bzw. der Virtual Satellite-Schnittstelle. Der Median der Messwerte der Bearbeitungszeit liegt für die HoloLens-Schnittstelle bei 87,5 Sekunden. Die 25%- und 75%-Quartile liegen bei 57,25 und 180 Sekunden. Die 10%- und 90%-Quartile liegen bei 40 und 267 Sekunden. Für die Messungen mit der Virtual Satellite-Schnittstelle liegt der Median 131,5 Sekunden. Die 25%- und 75%-Quartile liegen bei 96,25 und 165,75 Sekunden. Die 10%- und 90%-Quartile liegen bei 83 und 177 Sekunden. Mit dem Paired t-Test kann die statistische Signifikanz auch für Task C nicht nachgewiesen werden. Es ergibt sich ein p-Wert von $0,1953$. Dieser Wert liegt über dem geforderten Signifikanzniveau $\alpha = 0,05$. Die Nullhypothese H_0 kann somit für Task C nicht verworfen werden.

Schnittstelle	Messwert	Kritischer Wert	Teststatistik
HoloLens	Präz. Pos. Gesamt	0,226 $(n = 36)$	$0,3695$
HoloLens	Präz. Pos. Task A	0,375 $(n = 12)$	$0,1865$
HoloLens	Präz. Pos. Task B	0,375 $(n = 12)$	$0,1158$
HoloLens	Präz. Pos. Task C	0,375 $(n = 12)$	$0,3146$
HoloLens	Präz. Rot. Gesamt	0,226 $(n = 36)$	$0,1936$
HoloLens	Präz. Rot. Task A	0,375 $(n = 12)$	$0,1882$
HoloLens	Präz. Rot. Task B	0,375 $(n = 12)$	$0,1903$
HoloLens	Präz. Rot. Task C	0,375 $(n = 12)$	$0,2378$
Virtual Satellite	Präz. Pos. Gesamt	0,226 $(n = 36)$	$0,3546$
Virtual Satellite	Präz. Pos. Task A	0,375 $(n = 12)$	$0,3544$
Virtual Satellite	Präz. Pos. Task B	0,375 $(n = 12)$	$0,4465$
Virtual Satellite	Präz. Pos. Task C	0,375 $(n = 12)$	$0,3589$
Virtual Satellite	Präz. Rot. Gesamt	0,226 $(n = 36)$	$0,4566$
Virtual Satellite	Präz. Rot. Task A	0,375 $(n = 12)$	$0,3862$
Virtual Satellite	Präz. Rot. Task B	0,375 $(n = 12)$	$0,4033$
Virtual Satellite	Präz. Rot. Task C	0,375 $(n = 12)$	$0,4102$

Tabelle 4.3: Teststatistiken für die in der Nutzerstudie erhobenen Messwerte der Positions- und Rotationsabweichungen.

Die Kolmogorow-Smirnov-Teststatistiken für die Positions- und Rotationsabweichungen können aus Tabelle 4.3 entnommen werden. Für die HoloLens-Schnittstelle überschreiten die Messwerte der Positionspräzision über alle Tasks den kritischen Wert von $0,226$ und sind damit nach Maßgabe des Kolmogorow-Smirnov-Tests für ein Signifikanzniveau von $\alpha = 0,05$ nicht normalverteilt. Die Teststatistiken der Positions- und Rotationspräzision für Task A, Task B und Task B unterschreiten ihre entsprechenden kritischen Werte jedoch, womit die Nullhypothese nicht verworfen werden kann (Stichproben sind normalverteilt). Auch die Messwerte der Rotationspräzision über alle Tasks sind normalverteilt. Für die Virtual Satellite-Schnittstelle überschreiten sowohl die Messwerte der Positions- und Rotationspräzision den kritischen Wert von $0,226$. Die Positionspräzision für Task A und Task B ist dagegen normalverteilt. Die Positionspräzision für Task B, sowie die Rotationspräzision für Task A, Task B und Task C sind hingegen nicht normalverteilt. Dieses uneindeutige Ergebnis verkompliziert die Auswertung der Messdaten, da nicht in allen Fällen ein Paired t-Test verwendet werden kann. Da es sich bei allen Messdaten jedoch um rationale Werte handelt, kann für die Signifikanzanalyse ein Wilcoxon-Vorzeichen-Rang-Test verwendet werden [49].

Das Signifikanzniveau für den Wilcoxon-Vorzeichen-Rang-Test wird auf $\alpha = 0,05$ festgelegt. Für $n = 36$ Messwerte (alle Tasks) ergibt sich ein kritischer Wert von 208 für die Teststatistik. Für $n = 12$ Messwerte (Messungen pro Task) ergibt sich ein kritischer Wert von 13 für die Teststatistik. Es wird für die Positionsabweichung eine Nullhypothese $H_{0,Pos}$ und eine Alternativhypothese $H_{A,Pos}$ formuliert:

$H_{0,Pos}$: Der Unterschied zwischen den Positionsabweichungen bei den Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch nicht signifikant.

$H_{A,Pos}$: Der Unterschied zwischen den Positionsabweichungen bei den Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch signifikant.

Analog wird für die Rotationsabweichungen eine Null Nullhypothese $H_{0,Rot}$: und eine Alternativhypothese $H_{A,Rot}$ formuliert:

$H_{0,Rot}$: Der Unterschied zwischen den Rotationssabweichungen bei den Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch nicht signifikant.

$H_{A,Rot}$: Der Unterschied zwischen den Rotationsabweichungen bei den Tasks mit der Virtual Satellite und der HoloLens-Schnittstelle ist statistisch signifikant.

Für Teststatistiken unterhalb des kritischen Werts wird die Nullhypothese $H_{0,Pos}$ bzw. $H_{0,Rot}$ verworfen und die Alternativhypothese $H_{A,Pos}$ bzw. $H_{A,Rot}$ angenommen.

Das Balkendiagramm in Abbildung 4.15 zeigt die Mittelwerte der Abweichungen von der Zielposition für die HoloLens- und Virtual Satellite-Schnittstelle. Für die HoloLens-Schnittstelle beträgt die Abweichung im Mittel 2,717 Zentimeter. Für die Virtual Satellite-Schnittstelle im Mittel 3,003 Zentimeter. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 322. Dies liegt oberhalb des kritischen Werts

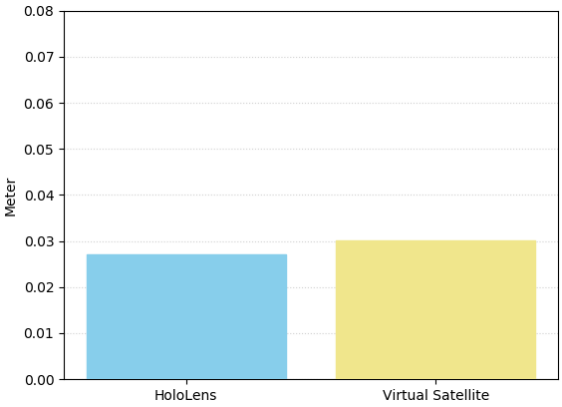


Abbildung 4.15: Arithmetisches Mittel der Positionsabweichungen über alle Tasks: HoloLens (blau) und Virtual Satellite (gelb).

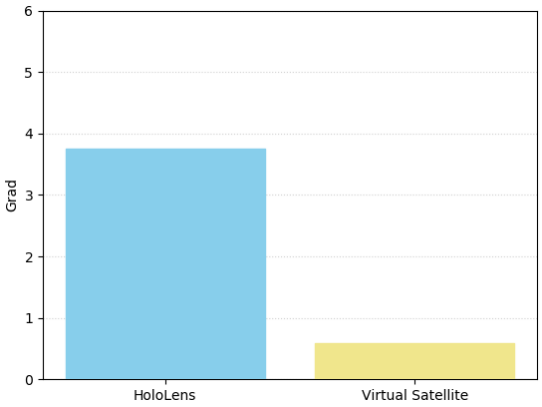


Abbildung 4.16: Arithmetisches Mittel der Rotationsabweichungen über alle Tasks: HoloLens (blau) und Virtual Satellite (gelb).

von 208 für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Pos}$ kann entsprechend nicht verworfen werden. Das Balkendiagramm in Abbildung 4.16 zeigt entsprechend die Mittelwerte der Abweichungen von der Zielrotation für die HoloLens- und Virtual Satellite-Schnittstelle. Für die HoloLens-Schnittstelle beträgt die Abweichung im Mittel 3,755 Grad. Für die Virtual Satellite-Schnittstelle im Mittel 0,5882 Grad. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt in diesem Fall 27. Dies liegt unterhalb des kritischen Werts von 208 für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Rot}$ kann in diesem Fall entsprechend verworfen werden.

Durch individuelle Betrachtung der einzelnen Tasks kann dieses Ergebnis ebenfalls differenziert werden. Im folgenden werden daher, die Positions- und Rotationsabweichungen für jeden Task separat ausgewertet ($n = 12$). Die sich stellende Frage ist, ob es Unterschiede zwischen den einzelnen Tasks bei den Positions- und Rotationsabweichungen gibt.

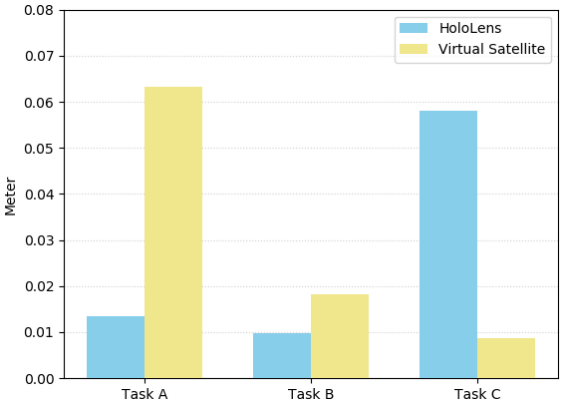


Abbildung 4.17: Mittelwerte der Positionsabweichungen für Task A, B und C: HoloLens (blau) und Virtual Satellite (gelb).

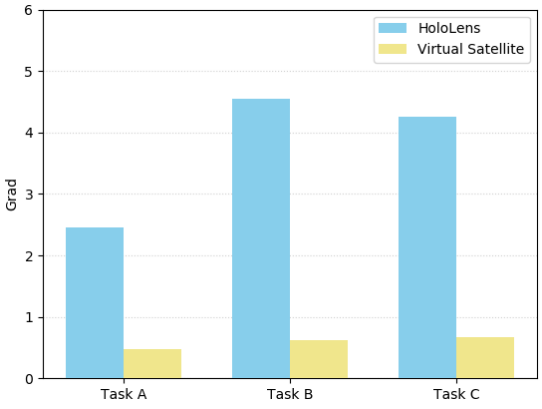


Abbildung 4.18: Mittelwerte der Rotationsabweichungen für Task A, B und C: HoloLens (blau) und Virtual Satellite (gelb).

Im Balkendiagramm 4.17 sind die Mittelwerte der Positionsabweichung für Task A, Task B und Task C ablesbar. Für Task A beträgt die Abweichung mit der HoloLens-Schnittstelle im Mittel 1,353 Zentimeter und mit der Virtual Satellite-Schnittstelle 6,326 Zentimeter. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test

beträgt 13. Dies ist genau der kritische Wert für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Pos}$ wird somit verworfen. Im Balkendiagramm 4.18 sind die Mittelwerte der Rotationsabweichung für Task A, Task B und Task C ablesbar. Für Task A beträgt die Abweichung mit der HoloLens-Schnittstelle im Mittel 2,457 Grad. Mit der Virtual Satellite-Schnittstelle beträgt sie im Mittel 0,479 Grad. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 9. Dies liegt unterhalb des kritischen Werts von 13 für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Rot}$ wird verworfen und die Alternativhypothese $H_{A,Rot}$ angenommen.

Für Task B beträgt die Positionsabweichung mit der HoloLens-Schnittstelle im Mittel 0,987 Zentimeter und mit der Virtual Satellite-Schnittstelle 1,814 Zentimeter. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 12. Dies liegt unter dem kritischen Wert von 13 für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Pos}$ wird somit verworfen. Die Rotationsabweichung mit der HoloLens-Schnittstelle liegt im Mittel bei 4,547 Grad. Mit der Virtual Satellite-Schnittstelle beträgt sie im Mittel 0,618 Grad. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 0. Die Nullhypothese $H_{0,Rot}$ wird somit für Task B verworfen.

Für Task C beträgt die Positionsabweichung mit der HoloLens-Schnittstelle im Mittel 5,811 Zentimeter. Mit der Virtual Satellite-Schnittstelle liegt die Abweichung im Mittel bei 0,868 Zentimetern. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 24. Dies liegt über dem kritischen Wert 13 für das geforderte Signifikanzniveau von $\alpha = 0,05$. Die Nullhypothese $H_{0,Pos}$ wird kann somit nicht verworfen werden. Die Rotationsabweichung mit der HoloLens-Schnittstelle liegt im Mittel bei 4,261 Grad. Mit der Virtual Satellite-Schnittstelle beträgt sie im Mittel 0,667 Grad. Die Teststatistik des Wilcoxon-Vorzeichen-Rang-Test beträgt 1. Die Nullhypothese $H_{0,Rot}$ wird somit für die Rotationsabweichungen von Task C ebenfalls verworfen.

Auf die Frage „How mentally demanding was it to perform the tasks?“ konnten sich die Probanden auf einer Skala von 1 (Very Low) bis 7 (Very High) einordnen. Für die Virtual Satellite-Schnittstelle gaben die Probanden im Mittel den Wert 4 an und für die HoloLens-Schnittstelle den Wert 3,25. Auf die Frage „How physically demanding was it to perform the tasks?“ wurde auf einer Skala von 1 (Very Low) bis 7 (Very High) im Mittel der Wert 1,9 (Virtual Satellite) und 3,8 (HoloLens) angegeben. Auf die Frage „How successful were you in accomplishing the tasks quickly?“ gaben die Probanden auf einer Skala von 1 (Perfect) bis 7 (Failure) im Mittel 2,25 für die VIrtual Satellite- und 3,2 für die HoloLens-Schnittstelle an. Für die Frage „How successful were you in accomplishing the tasks quickly?“ wurde im Mittel auf einer Skala von 1 (Perfect) bis 7 (Failure) der Wert 4 (Virtual Satellite) und 3,9 (HoloLens) angegeben. Für die Frage „How successful were you in accomplishing the tasks with precision?“ gaben die Probanden auf einer Skala von 1 (Perfect) bis 7 (Failure) im Mittel den Wert 2,1 für die Virtual Satellite- und 3,8 für die HoloLens-Schnittstelle an. Für die Frage „How hard did you have to work to accomplish your level of performance?“ ergaben sich im Mittel die Werte 4 für die Virtual Satellite-Schnittstelle und 4,5 für die HoloLens-Schnittstelle. Für die Frage „How insecure, discouraged, irritated, stressed and annoyed were you?“ wurde im Mittel für beide Schnittstellen der Wert 3 angegeben.

4.3 Diskussion

Anhand der in Abschnitt 4.2 ermittelten Ergebnisse können Schlüsse über Vor- und Nachteile der HoloLens-Anwendung gegenüber der desktopbasierten Virtual Satellite-Schnittstelle gezogen werden.

In der Gesamtbetrachtung aller Tasks konnte ein statistisch signifikanter Zusammenhang zwischen der verwendeten Schnittstelle und der Durchführungszeit gefunden werden. Dieser Befund wird allerdings durch die Einzelbetrachtung der Tasks relativiert. Für Task A konnte ein statistisch signifikanter Zusammenhang zwischen Schnittstelle und Durchführungszeit gefunden werden, für Task B und Task C jedoch nicht.

Eine Erklärung für die im Vergleich zur HoloLens-Schnittstelle hohe Präzision bei der Orientierung ist schnell gefunden. Alle Tasks erforderten die Ausrichtung und Positionierung eines gelben Zylinders auf der Oberfläche eines würfelförmigen Zielobjekts. Die meisten Probanden haben für die Tests mit der Virtual Satellite-Schnitt- stelle die numerischen Rotationsparameter des Zielobjekts in die Textfelder für die Rotationsparameter des Zylinders kopiert. Damit war der Zylinder bereits perfekt ausgerichtet. Die selbe Strategie wurde häufig auch für die Positionsparameter des Zylinders verwendet. Da der Zylinder jedoch auf den Würfeloberflächen positioniert werden musste, erforderte dies eine zusätzliche Verschiebung des Zylinders. Manche Probanden versuchten die notwendige Verschiebung anhand der Ausdehnungsparameter des Würfels exakt zu berechnen, andere näherten sich der richtigen Verschiebung durch das Austesten verschiedener Positionsparameter. In beiden Fällen kostete dies viel Zeit.

Eine weitere interessante Beobachtung war, dass die Probanden die Rotationsmetapher häufiger und stärker bemängelten, wenn sie die Tasks zuerst mit der Virtual Satellite-Schnittstelle bearbeiteten. Der Verdacht liegt nahe, dass die Probanden versuchten ihre guten Ergebnisse bei der Ausrichtung mit der Virtual Satellite-Schnittstelle im zweiten Testteil mit der HoloLens-Schnittstelle ebenso gut zu wiederholen, was allerdings als besonders schwierig empfunden wurde. Probanden, welche die Tests zuerst mit der HoloLens-Schnittstelle durchführten, hatten ein positiveres Bild von der HoloLens-Schnittstelle.

Aus der Auswertung des Fragebogens geht hervor, dass die Probanden den mentalen Aufwand bei der HoloLens-Schnittstelle als geringer empfanden. Den physischen Aufwand empfanden sie dagegen als höher im Vergleich zu der Virtual Satellite-Schnittstelle. Diese Ergebnisse decken sich mit den Beobachtungen: Mit der Virtual Satellite-Schnittstelle wurde die Position und Orientierung teilweise sogar ausgerechnet. Dies erklärt unter anderem den erhöhten mentalen Aufwand. Bei Verwendung der HoloLens-Schnittstelle wurden häufig Ermüdungserscheinungen des Arms beklagt, der für die Manipulation der Objekte stets in der Luft gehalten werden musste. Dies erklärt die höhere physische Belastung für die Probanden.

Zusammengefasst kann festgehalten werden, dass die HoloLens-Anwendung im Hinblick auf Positionierungsaufgaben Vorteile gegenüber der Virtual Satellite-Schnittstelle hat. Diese Vorteile betreffen in erster Linie die niedrigere Durchführungszeit

von Docking- und Montageaufgaben. Hinsichtlich der Rotation virtueller Objekte hat die HoloLens-Anwendung bei der freien Manipulation jedoch starke Schwächen offenbart. Der größte Teil der Durchführungszeit wurde für die Ausrichtung des Zylinders investiert. Mit der Virtual Satellite-Schnittstelle vergleichbar niedrige Rotationsabweichungen wurden dabei jedoch verfehlt. Der Verfasser dieser Arbeit ist überzeugt, dass sich diese Probleme durch Verwendung der Snapping-Metapher weitgehend beheben lassen. Gleichzeitig muss jedoch auch die Rotationsmetapher für die freie Manipulation weiterentwickelt werden, um auch ohne Ausrichtungswerkzeuge eine akzeptable Präzision zu erreichen.

5 Schlussfolgerung und Ausblick

In dieser Arbeit wurde eine immersive AR-Anwendung für die Raumfahrzeugmontage entwickelt und evaluiert. Dazu wurden Anforderungen an die Raumfahrzeugmontage in immersiven, virtuellen Umgebungen untersucht und auf Grundlage dessen ein vollständiges Interaktionskonzept entworfen. Zudem wurde die Anwendung in die Client-Server-Architektur von Virtual Satellite eingebettet.

In Kapitel 4 wurde das Interaktionskonzept in einer Nutzerstudie evaluiert. Aus den Ergebnissen der Evaluation konnten in Abschnitt 4.3 Schlüsse über Vor- und Nachteile der entwickelten HoloLens-Anwendung gegenüber der Virtual Satellite Desktopschnittstelle für die Raumfahrzeugkonfiguration gezogen werden. Die Ergebnisse geben Anlass zu weiteren Untersuchungen. Die notwendige Weiterentwicklung der Manipulationsmetapher für die Rotation ist hier ein offensichtlicher Punkt. Ansatzpunkte dafür können bei Hinckley et al. [50]. In weiteren Nutzerstudien muss zudem geprüft werden, wie sich die Verwendung der implementierten Ausrichtungswerkzeuge auf die Durchführungsgeschwindigkeit, sowie die Positions- und Rotationspräzision auswirkt.

Insbesondere in einer Einrichtung wie der CEF sind auch Aspekte der kollaborativen bzw. kooperativen Objektmanipulation interessante Untersuchungsgegenstände. Die Einbettung der HoloLens-Anwendung in die bestehende Kommunikationsarchitektur von Virtual Satellite hat hierfür die Grundlage geschaffen. Für zukünftige Iterationsschritte ist die Erweiterung der HoloLens-Anwendung um kollaborative Manipulationsmetaphern naheliegend. In der wissenschaftlichen Literatur lassen sich dazu bei Pinho et al und Otto et al. Ansätze finden [51, 52].

Zusammenfassend kann festgehalten werden, dass sich ausgehend von der in dieser Arbeit vorgestellten HoloLens-Anwendung in unterschiedlichen Richtungen produktive Untersuchungsgegenstände auftun. Schließlich können Fortschritte im Bereich immersiver AR-Technologien völlig neue Möglichkeiten schaffen, die nicht nur für die Raumfahrzeugmontage Verbesserungspotentiale bergen. Es darf angenommen werden, dass sich Untersuchungen und Entwicklungen in diesem Bereich auch in Zukunft als ergiebig erweisen.

Literatur

[1] Holger Schumann u. a. „Overview of the new Concurrent Engineering Facility at DLR“. In: *3rd International Workshop on System & Concurrent Engineering for Space Applications (SECESA)*. 2008.

[2] Philipp M. Fischer, Robin Wolff und Andreas Gerndt. „Collaborative satellite configuration supported by interactive visualization“. In: *2012 IEEE Aerospace Conference*. 2012, S. 1–11.

[3] Holger Schumann u. a. „Concurrent Systems Engineering in Aerospace: From Excel-based to Model Driven Design“. In: *8th Conference on Systems Engineering Research*. März 2010.

[4] Meenakshi Deshmukh u. a. „Interactive 3D Visualization to Support Concurrent Engineering in the Early Space Mission Design Phase“. In: *Challenges in European Aerospace*. Okt. 2015.

[5] Colin Ware, Kevin Arthur und Kellogg S. Booth. „Fish Tank Virtual Reality“. In: *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*. CHI '93. ACM, 1993, S. 37–42.

[6] Microsoft Corporation. *Microsoft HoloLens*. URL: `https://www.microsoft.com/de-de/hololens` (besucht am 21. 03. 2018).

[7] Shimpali Deshpande und Geeta Uplenchwar. „Google Glass“. In: *International Journal of Scientific & Engineering Research*. Bd. 4. Dez. 2013.

[8] David M. Ewalt. *Inside Magic Leap, The Secretive $4.5 Billion Startup Changing Computing Forever*. Nov. 2016. URL: `https://www.forbes.com/sites/davidewalt/2016/11/02/inside-magic-leap-the-secretive-4-5-billion-startup-changing-computing-forever` (besucht am 21. 03. 2018).

[9] Tilman Wittenhorst. *Intel zeigt Datenbrille Vaunt mit Netzhaut-Projektor*. Feb. 2018. URL: `https://www.heise.de/newsticker/meldung/Intel-zeigt-Datenbrille-Vaunt-mit-Netzhaut-Projektor-3960919.html` (besucht am 21. 03. 2018).

[10] Doug A. Bowman u. a. *3D User Interfaces: Theory and Practice*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004, S. 250–393.

[11] Wolfgang Birkfellner u. a. „A head-mounted operating binocular for augmented reality visualization in medicine - design and initial evaluation“. In: *IEEE Transactions on Medical Imaging* 21.8 (Aug. 2002), S. 991–997.

[12] Xiaohui Zhang, Nassir Navab und Shih-Ping Liou. „E-commerce direct marketing using augmented reality“. In: *2000 IEEE International Conference on Multimedia and Expo. ICME2000. Latest Advances in the Fast Changing World of Multimedia*. Bd. 1. 2000, S. 88–91.

[13] Paul Milgram u. a. „Augmented reality: A class of displays on the reality-virtuality continuum“. In: *Telemanipulator and Telepresence Technologies (1994) – SPIE Vol. 2351*. International Society for Optics und Photonics. 1994, S. 282–292.

[14] Robert J.K. Jacob u. a. „Reality-based Interaction: A Framework for post-WIMP Interfaces“. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. Florence, Italy: ACM, 2008, S. 201–210.

[15] Andries van Dam. „Post-WIMP User Interfaces“. In: *Commun. ACM* 40.2 (Feb. 1997), S. 63–67.

[16] Chris Angelini. *Microsoft HoloLens: HPU Architecture, Detailed*. 2016. URL: `http://www.tomshardware.com/news/microsoft-hololens-hpu-architecure-28nm,32586.html` (besucht am 20. 01. 2018).

[17] Windows Dev Center. *HoloLens hardware details*. URL: `https://developer.microsoft.com/en-us/windows/mixed-reality/hololens_hardware_details` (besucht am 04. 02. 2018).

[18] Bernard C. Kress und William J. Cummings. „Optical architecture of HoloLens mixed reality headset“. In: *SPIE Digital Optical Technologies*. Bd. 10335. 2017.

[19] Seth Colaner. *What's Inside Microsoft's HoloLens And How It Works*. 2016. URL: `http://www.tomshardware.com/news/microsoft-hololens-components-hpu-28nm,32546.html` (besucht am 20. 01. 2018).

[20] Windows Dev Center. *Gestures*. URL: `https://docs.microsoft.com/en-us/windows/mixed-reality/gestures` (besucht am 22. 03. 2018).

[21] Stephen Hodgson et al. *MixedRealityToolkit-Unity, Input Readme*. Okt. 2017. URL: `https://github.com/Microsoft/MixedRealityToolkit-Unity/blob/master/Assets/HoloToolkit/Input/README.md` (besucht am 22. 03. 2018).

[22] Windows Dev Center. *Spatial mapping*. URL: `https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_mapping` (besucht am 21. 01. 2018).

[23] Holger Schumann u. a. „DLR's Virtual Satellite Approach“. In: *10th International Workshop on Simulation on European Space Programmes (SESP 2008)*. Okt. 2008.

[24] Pieter Hintjens. *ØMQ - The Guide*. URL: `http://zguide.zeromq.org/page:all` (besucht am 23. 03. 2018).

[25] Gabriel Gambetta. *Fast-Paced Multiplayer*. 2017. URL: `http://www.gabrielgambetta.com/client-server-game-architecture.html` (besucht am 23. 03. 2018).

[26] European Cooperation for Space Standardization. *ECSS–M–ST–10C Rev. 1 - Space project management - Project planning and implementation*. 2009.

[27] Doug A. Bowman u. a. *3D User Interfaces: Theory and Practice*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004, S. 360–393.

[28] Andrew Sears und Julie A. Jacko. *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications (Human Factors and Ergonomics Series)*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 2007, S. 148–152.

[29] Andrew Wilson und Nuria Oliver. *Multimodal Sensing for Explicit and Implicit Interaction*. Redmond, WA, USA: Microsoft Research.

[30] Alan J. Dix u. a. „Intelligent context-sensitive interactions on desktop and the web“. In: *CAI*. 2006.

[31] Paul Adamczyk. „The anthology of the finite state machine design patterns“. In: (März 2018).

[32] Doug A. Bowman u. a. *3D User Interfaces: Theory and Practice*. Redwood City, CA, USA: Addison Wesley Longman Publishing Co., Inc., 2004, S. 252–305.

[33] Ralf Dörner u. a. *Virtual und Augmented Reality (VR/AR)*. eXamen.press. Springer Vieweg, 2013, S. 160–165.

[34] Robert J. Teather. „Evaluating 3D Pointing Techniques“. Diss. Toronto, Ontario, Canada: York University, 2013.

[35] Marcus Tönnis. *Augmented Reality. Einblicke in die Erweiterte Realität*. Informatik im Fokus. Springer, 2010, S. 96–106.

[36] Colin Ware und Kathy Lowther. „Selection Using a One-eyed Cursor in a Fish Tank VR Environment“. In: *ACM Trans. Comput.-Hum. Interact.* 4.4 (1997), S. 309–322.

[37] Ivan Poupyrev u. a. „The Go-go Interaction Technique: Non-linear Mapping for Direct Manipulation in VR“. In: *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology*. UIST '96. Seattle, Washington, USA: ACM, 1996, S. 79–80.

[38] Scott Frees, G. Drew Kessler und Edwin Kay. „PRISM Interaction for Enhancing Control in Immersive Virtual Environments“. In: *ACM Trans. Comput.-Hum. Interact.* 14.1 (2007).

[39] Tovi Grossman und Ravin Balakrishnan. „The Design and Evaluation of Selection Techniques for 3D Volumetric Displays“. In: *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. UIST '06. Montreux, Switzerland: ACM, 2006, S. 3–12.

[40] Lode Vanacken, Tovi Grossman und Karin Coninx. „Multimodal Selection Techniques for Dense and Occluded 3D Virtual Environments“. In: *Int. J. Hum.-Comput. Stud.* 67.3 (März 2009), S. 237–255.

[41] Felipe Bacim, Regis Kopper und Doug A. Bowman. „Design and evaluation of 3D selection techniques based on progressive refinement“. In: *International Journal of Human-Computer Studies* 71.7 (2013), S. 785–802.

[42] Ivan Poupyrev u. a. „A Framework and Testbed for Studying Manipulation Techniques for Immersive VR“. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. VRST '97. Lausanne, Switzerland: ACM, 1997, S. 21–28.

[43] Shumin Zhai. „User Performance in Relation to 3D Input Device Design“. In: *SIGGRAPH Comput. Graph.* 32.4 (1998), S. 50–54.

[44] Shumin Zhai und Paul Milgram. „Quantifying Coordination in Multiple DOF Movement and Its Application to Evaluating 6 DOF Input Devices“. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '98. Los Angeles, California, USA: ACM Press/Addison-Wesley Publishing Co., 1998, S. 320–327.

[45] Colin Ware. „Using hand position for virtual object placement“. In: *The Visual Computer* 6.5 (Sep. 1990), S. 245–253.

[46] Lawrence M. Parsons. „Inability to reason about an object's orientation using an axis and angle of rotation“. In: *Journal of Experimental Psychology: Human Perception and Performance* 21.6 (1995), S. 1259–1277.

[47] Jens Herder und Thomas Novotny. „Spatial Sound Design and Interaction for Virtual Environments in the Promotion of Architectural Designs“. In: *Third International Workshop on Spatial Media*. University of Aizu. Aizu-Wakamatsu, Japan, 2003, S. 7–11.

[48] Paul M. Fitts. „The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement“. In: 121 (Okt. 1992), S. 262–269.

[49] Paul Grimm u. a. „Virtual und Augmented Reality (VR/AR) :Grundlagen und Methoden der Virtuellen und Augmentierten Realität“. In: Hrsg. von Ralf Dörner u. a. eXamen.press. Springer, Jan. 2014, S. 183–191.

[50] Ken Hinckley u. a. „Usability Analysis of 3D Rotation Techniques“. In: *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*. UIST '97. Banff, Alberta, Canada: ACM, 1997, S. 1–10.

[51] Márcio S. Pinho, Doug A. Bowman und Carla M.D.S. Freitas. „Cooperative Object Manipulation in Immersive Virtual Environments: Framework and Techniques“. In: *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*. VRST '02. Hong Kong, China: ACM, 2002, S. 171–178.

[52] Oliver Otto, Dave Roberts und Robin Wolff. „A Review on Effective Closely-coupled Collaboration Using Immersive CVE's“. In: *Proceedings of the 2006 ACM International Conference on Virtual Reality Continuum and Its Applications*. VRCIA '06. Hong Kong, China: ACM, 2006, S. 145–154.

User Test Questionnaire

Participant	Interface	Date

---

1. *How mentally demanding was it to perform the tasks?*

Very Low Very High

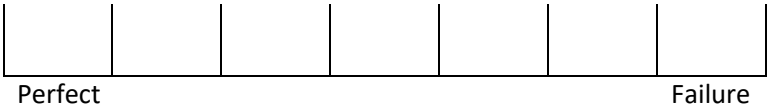
Comment:

2. *How physically demanding was it to perform the tasks?*

Very Low Very High

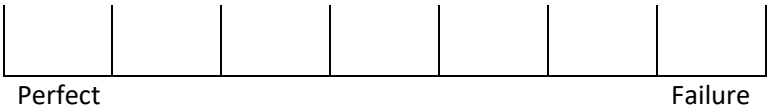
Comment:

3. *How successful were you in accomplishing what you were asked to do?*



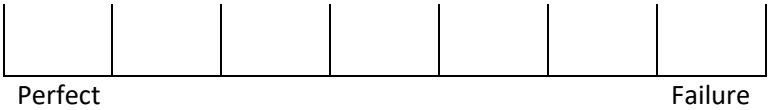
Comment:

4. *How successful were you in accomplishing the tasks quickly?*



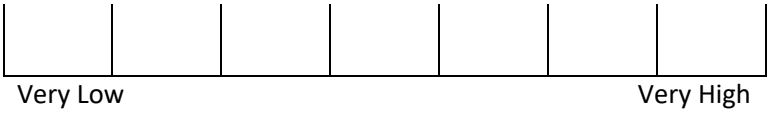
Comment:

5. *How successful were you in accomplishing the tasks with precision?*



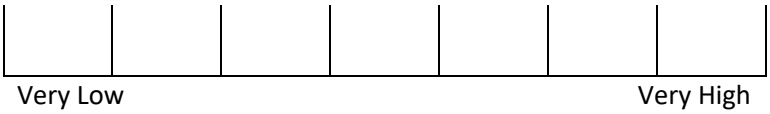
Comment:

6. *How hard did you have to work to accomplish your level of performance?*



Comment:

7. *How insecure, discouraged, irritated, stressed and annoyed were you?*



Comment:

Further remarks: